

DLV11J

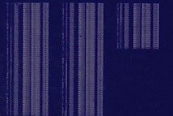
DLV11-J TEST
CNDLAAO

AH-T470A-MC
FICHE 1 OF 1

MAY 1983
COPYRIGHT © 82-83
MADE IN USA



Table with multiple columns and rows of data, including headers like 'TYPE', 'DESCRIPTION', 'SPECIFICATIONS', and 'REMARKS'. The content is highly repetitive and difficult to read due to the image quality.



.REM %

IDENTIFICATION

PRODUCT CODE: AC-T469A-MC
PRODUCT NAME: CNDLAAO DLV11-J TEST
PRODUCT DATE: DEC, 1982
MAINTAINER: DIAGNOSTIC SERVICES/ISS

AUTHOR: DIAG/ISS

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C)1982,1983 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

| | | | |
|---------|-------|---------|---------|
| DIGITAL | PDP | UNIBUS | MASSBUS |
| DEC | DECUS | DECTAPE | |

TABLE OF CONTENTS

| | |
|-----|-------------------------------------|
| 1.0 | GENERAL PROGRAM INFORMATION. |
| 1.1 | PROGRAM PURPOSE (ABSTRACT). |
| 1.2 | SYSTEM REQUIREMENTS. |
| 1.3 | RELATED DOCUMENTS AND STANDARDS. |
| 1.4 | DIAGNOSTIC HIERARCHY PREREQUISITES. |
| 1.5 | ASSUMPTIONS. |
| 2.0 | OPERATING INSTRUCTIONS. |
| 2.1 | LOADING AND STARTING PROCEDURES. |
| 2.2 | SPECIAL ENVIRONMENTS. |
| 2.3 | OPERATIONAL SWITCH SETTINGS |
| 2.4 | PROGRAM OPTIONS. |
| 2.5 | EXECUTION TIMES. |
| 2.6 | POWER FAIL. |
| 3.0 | ERROR INFORMATION. |
| 3.1 | ERROR REPORTING PROCEDURE. |
| 3.2 | ERROR HALTS. |
| 4.0 | PERFORMANCE AND PROGRESS REPORTS. |
| 4.1 | PERFORMANCE REPORTS. |
| 5.0 | DEVICE INFORMATION TABLES. |
| 6.0 | SUMMARY OF TESTS. |

1.0 GENERAL PROGRAM INFORMATION.

1.1 PROGRAM PURPOSE (ABSTRACT).

THIS DIAGNOSTIC IS A LOGIC TEST TO VERIFY THE OPERATION OF THE DLV11-J SERIAL INTERFACE.
TESTING IS DONE IN TWO DISTINCT PHASES:

1. ALL SELECTED CHANNELS PER DLV11-J MODULE ARE TESTED INDIVIDUALLY
2. THE DLV11-J MODULE IS TESTED AS A WHOLE FOR CHANNEL INTERACTION PROBLEMS. THIS DIAGNOSTIC IS DESIGNED TO TEST AND DETECT ERRORS TO THE LOGIC LEVEL (NOT TO THE CHIP LEVEL).

THIS DIAGNOSTIC OPERATES UP TO 2 DLV11-J SERIAL LINE INTERFACES CONFIGURED AT CONSECUTIVE BASE ADDRESSES.
THE PROGRAM WILL DO AUTO-SIZING IF THE DEVICE MAP '\$DEVN' = 0.
ALTERNATELY, THE OPERATOR CAN SELECT OR DESELECT INDIVIDUAL MODULES & CHANNELS BY SETTING THE PROPER BITS IN '\$DEVN' (SEE PROGRAM OPTIONS SEC. 2.4) BEFORE RUNNING THE PROGRAM.
BY THIS METHOD, THE PROGRAM WILL BYPASS AUTO-SIZING AND TEST ONLY THOSE MODULES/CHANNELS IT FINDS IN \$DEVN.

IN EITHER CASE, THE PROGRAM WILL PRINT OUT ALL MODULES & CHANNELS TO BE TESTED BEFORE PROCEEDING.

THE OPERATOR MUST INSTALL DATA WRAP AROUND CONNECTORS TO DO DATA TESTING. TO BYPASS DATA TESTS, THE OPERATOR MUST MODIFY '\$USWR' (USER SWITCH REGISTER), SEE PROGRAM OPTIONS SEC. 2.4)

THE DEFAULT ADDRESSES & VECTORS ARE AS FOLLOWS:

177560 -CONSOLE INTERFACE DEVICE ADDRESS
176500 -FIRST SERIAL CHANNEL ADDRESS OF UP TO 8 CONSECUTIVE SERIAL LINE DEVICES.

60 - VECTOR FOR CONSOLE DEVICE INTERFACE.
300 - VECTOR FOR FIRST OF 16 DEVICES.

THIS PROGRAM IS DESIGNED TO RUN ON ANY Q-BUS PDP-11 WITH 4K OF MEMORY AND A DLV11-J (Q-BUS) MODULE. IT CAN RUN UNDER XXDP & APT MONITORS, AND ON PROCESSORS WITH NO HARDWARE SWITCH REGISTER.

1.2 SYSTEM REQUIREMENTS.

HARDWARE REQUIREMENTS:

ANY Q-BUS PDP-11 FAMILY PROCESSOR
4K MEMORY - MINIMUM
A SPECIAL DATA WRAP AROUND CONNECTOR OR EQUIVALENT
(REQ'D IF DATA WRAP AROUND TESTS DESIRED)

IF CHANNEL 3 IS CONFIGURED AS THE CONSOLE:
TESTS 6-12, 14-17, 21, 22 ARE BYPASSED.

IF DATA WRAP AROUND TESTS ARE BYPASSED:
TESTS 7-12, 14-17, 21, 22 ARE BYPASSED.

IF CHANNEL 3 IS CONFIGURED AS THE CONSOLE & A VT-100 IS
THE CONSOLE DEVICE, SWITCH THE VT-100 TO "JUMP SCROLL" MODE
TO AVOID LOOSING CHARACTERS ON THE SCREEN.

SOFTWARE REQUIREMENTS:

THIS DIAGNOSTIC IS DESIGNED TO RUN IN ANY OF THE
FOLLOWING WAYS:

STAND ALONE
WITH APT MONITOR
WITH XXDP MONITOR (CHAINABLE IF RENAMED TO .BIC EXTENSION)

THIS DIAGNOSTIC IS NOT DESIGNED TO RUN WITH THE
DIAGNOSTIC SUPERVISOR.

1.3 RELATED DOCUMENTS AND STANDARDS.

| | |
|--|----------------|
| DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS | 175-003-009-02 |
| APT | MD-11-DZZMA |
| SYSMAC | MD-11-DZQAC |

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES.

NO SPECIAL DIAGNOSTICS ARE REQUIRED TO RUN BEFORE THIS, BUT
THE PROCESSOR, MEMORY, AND BUS ARE ASSUMED TO BE FULLY
OPERATIONAL.

1.5 ASSUMPTIONS.

THIS DIAGNOSTIC ASSUMES THAT THE OPERATOR HAS INITIALIZED
LOCATION '\$USWR' AND '\$DEVM' TO THE PROPER VALUES. (SEE SEC. 2.4)

2.0 OPERATING INSTRUCTIONS.

2.1 LOADING AND STARTING PROCEDURES.

USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED MEDIA.

THERE ARE 2 STARTING LOCATIONS FOR THIS DIAGNOSTIC:

1. LOCATION 200 FOR ALL NORMAL TESTING STARTS & RESTARTS.
2. LOCATION 400 FOR INTERRUPT VECTOR DEBUG... OFFLINE ONLY.

THE 400 START SHOULD BE USED ONLY TO DEBUG FAULTY VECTOR ADDRESSES PUT OUT BY THE DLV11-J DURING INTERRUPTS. STARTING AT 400 WILL CAUSE THE ENTIRE VECTOR AREA, EXCEPT THAT NEEDED BY SYSMAC, TO BE OVERWRITTEN & TO POINT TO A COMMON SERVICE ROUTINE TO ENABLE LOOPING ON THE PROBLEM.

AS SOON AS TESTING STARTS, THE OPERATOR CAN CHANGE THE SWITCH REGISTER ONLY BY A 'BREAK' & MANUALLY LOADING LOCATION 176 (SWREG) WITH THE DESIRED CONTENTS (SEE SEC. 2.3) THEN DOING A 'P' TO PROCEED.

IT IS IMPORTANT THAT WHEN GOING BACK TO NORMAL TESTING A FRESH LOAD OF THE PROGRAM BE PERFORMED TO RESET THE VECTOR AREA.

THE USER CAN SELECT A SPECIFIC TEST TO BE EXECUTED BY SETTING BIT 8 IN SWREG AND THE TEST NUMBER (IN OCTAL) IN BITS <7:0>.

(NOTE: ALL TESTS PREVIOUS TO THE SELECTED ONE ARE EXECUTED WITHOUT ITERATIONS.)

2.2 SPECIAL ENVIRONMENTS.

THIS DIAGNOSTIC FOLLOWS THE STANDARD PROCEDURE FOR RUNNING UNDER APT,XXDP MONITORS, AS DESCRIBED IN THEIR RESPECTIVE PROCEDURES MANUAL AND SYSMAC PACKAGE.

2.3 OPERATIONAL SWITCH SETTINGS

THE SOFTWARE SWITCH REGISTER (LOC. 176) IS USED FOR ALL OPERATIONAL SWITCH SETTINGS. THIS CAN BE ACCOMPLISHED IN THE FOLLOWING WAY:

- 1) TYPE CONTROL G <^G>: THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.
- 2) THE MACHINE WILL THEN TYPE: ' SWR=XXXXXX NEW=' (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)
- 3) AFTER THE 'NEW=' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:
 - A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED). LEADING ZEROS NEED NOT BE TYPED, AND IF MORE THAN 6 DIGITS ARE TYPED THE LAST 6 WILL BE USED. IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.
 - B) IF A CONTROL U <^U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 3.
 - C) IF THE INPUT CHARACTER IS NOT ONE OF THE CHARACTERS MENTIONED ABOVE THEN A QUESTION MARK (?) WILL BE TYPED FOLLOWED BY A CARRAGE RETURN AND A LINE FEED SEQUENCE THEN PROCEED FROM STEP 3 (ERASING ALL PREVIOUS INPUT).
- 4) THE DIAGNOSTIC WILL CONTINUE ON TYPING <CR>.

NOTE: BECAUSE OF FREQUENT BUS RESETS IN THE PROGRAM, IT MAY BE NECESSARY TO DO 'CONTROL-G' SEVERAL TIMES. ALTERNATELY, A 'BREAK' & MANUALLY LOADING LOC. 176, FOLLOWED BY A 'P' TO PROCEED WILL ALSO WORK.

SOFTWARE SWITCH REGISTER OPTIONS (SWREG)

BIT 15 SET = 100000 = HALT ON ERROR
14 SET = 40000 = LOOP ON TEST (TO BE USED ONLY WHILE TESTING IN PROGRESS)
13 SET = 20000 = INHIBIT ERROR TIMEOUTS
12 SET = 10000 = ENABLE PERFORMANCE REPORTS
11 SET = 4000 = INHIBIT ITERATIONS
10 SET = 2000 = BELL ON ERROR
9 SET = 1000 = LOOP ON ERROR
8 SET = 400 = LOOP ON TEST IN SWR<7:0>
7:0 = NUMBER OF TEST TO LOOP ON (USED WITH BIT 8)
(ALL TESTS PREVIOUS TO THE SELECTED TEST
ARE EXECUTED FIRST WITH 1 ITERATION ONLY)

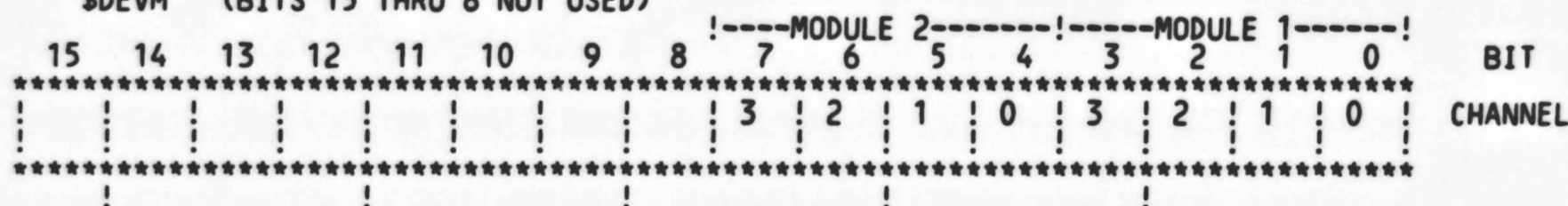
2.4 PROGRAM OPTIONS.

THIS PROGRAM REQUIRES THE ADDRESS OF THE FIRST RCSR (STORED AT '\$BASE') AND ITS INTERRUPT VECTOR (STORED AT '\$VECT1'). IT WILL BE ABLE TO ADDRESS ANY DLV11-J STARTING AT THE SPECIFIED BASE ADDRESS THRU 2 CONSECUTIVE MODULES.

EXAMPLES: \$BASE: 176500 (DEFAULT)
 \$VECT1: 300 (DEFAULT)

LOCATION '\$DEVN' IS USED AS A BIT MAP TO INDICATE WHICH UNIT NUMBERS ARE PRESENT AND WILL BE TESTED.

\$DEVN (BITS 15 THRU 8 NOT USED)



A BIT MAP CAN BE ENTERED AT '\$DEVN' PRIOR TO STARTING THE PROGRAM. IF '\$DEVN' IS LEFT AT ITS DEFAULT VALUE (0), THE PROGRAM WILL DO AUTO-SIZING BEGINNING AT THE DEVICE ADDRESS IN '\$BASE' & WILL SETUP '\$DEVN' ACCORDINGLY. IN EITHER CASE, THE PROGRAM WILL TYPE OUT A LIST OF ALL THE MODULES/CHANNELS TO BE TESTED BEFORE PROCEEDING.

NOTE: SIZING IS PERFORMED ONLY ONCE AT THE BEGINNING OF THE PROGRAM. IF \$DEVN IS TO BE CHANGED, THE PROGRAM MUST BE RESTARTED AT 200 FOR IT TO BE EFFECTIVE. OTHERWISE, \$DEVN WILL BE IGNORED.

OPTIONS

LOCATION \$USWR CONTAINS ALL THE USER SELECTABLE OPTIONS. THE VALUES IN THIS WORD MUST CONFORM TO THE ACTUAL BOARD CONFIGURATION.

| BITS | BIT POSITION | DEFINITION | \$USWR | DEFAULT VALUE |
|------|--------------|--|--------|--------------------------------------|
| | 0 | # OF DATA BITS TRANSMITTED 0 = 7 BITS, 1 = 8 BITS | 1 | = 8 BITS |
| | 1 | PARITY ENABLED | 0 | = NO |
| | 2 | EVEN ODD PARITY | 0 | = ODD |
| | 3 | BREAK DETECTION ENABLED | 1 | = YES |
| | 4 | RUN DATA WRAP AROUND TESTS | 1 | = YES |
| | <11:9> | CONSOLE DEVICE | 1 | = YES = CONSOLE ON MODULE 1, CHAN. 3 |

<11:9> = 0 = CONSOLE NOT ON DLV11-J
 <11:9> = 1 = CONSOLE ON MODULE 1
 <11:9> = 2 = CONSOLE ON MODULE 2

IMPORTANT:

1. FOR DIAGNOSTIC PURPOSES, ALL CHANNELS MUST BE CONFIGURED TO THE SAME BIT/WORD LENGTH.
2. THE 'SIZING' ROUTINE WILL SET THE APPROPRIATE CHAN 3 BIT IN \$DEVN IF BITS <11:9> ARE SET IN \$USWR.
3. IF RUNNING UNDER APT & THE CONSOLE IS ON THE DLV11-J, THE CONSOLE MUST NOT BE TESTED. THIS IS BECAUSE APT SENDS 'BREAKS' TO THE CONSOLE WHICH INTEREFERES WITH THE TESTS.

FOR THIS CASE:

- A. ALLOW THE \$DEVN TO STAY EITHER AT ITS DEFAULT OF 0 OR CLEAR THE APPROPRIATE CHAN. 3 BIT.
- B. CLEAR BITS <11:9> IN \$USWR SO THE CONSOLE BIT WILL NOT BE SET INTO \$DEVN FOR TESTING.

SUMMARY OF DEVICE ADDRESSES & VECTORS.

| MODULE # | CHANNEL | DEV ADDRESS (RCSR) | VECTORS | |
|----------------|---------|-----------------------|------------|-----------|
| | | | REC | XMIT |
| MODULE #1 | 0 | \$BASE+00 | \$VECT1+00 | \$VECT1+4 |
| | 1 | +10 | +10 | +14 |
| | 2 | +20 | +20 | +24 |
| MODULE #2 | 0 | +30 | +30 | +34 |
| | 1 | +40 | +40 | +44 |
| | 2 | +50 | +50 | +54 |
| | 3 | +60 | +60 | +64 |
| | | +70 | +70 | +74 |
| CONSOLE DEVICE | 3 | \$TKS | TKVEC=60 | TPVEC=64 |

SUMMARY OF USER LOCATIONS & DEFAULTS.

| | LOC | DEFAULT | |
|---------|------|---------|---|
| \$BASE | 1250 | 176500 | |
| \$VECT1 | 1244 | 300 | |
| \$TKS | 1144 | 177560 | FOR CONSOLE DEVICE ON MODULE UNDER TEST |
| TKVEC | | 60 | FOR CONSOLE DEVICE ON MODULE UNDER TEST |
| TPVEC | | 64 | FOR CONSOLE DEVICE ON MODULE UNDER TEST |
| \$USWR | 1220 | 31 | SEE SEC. 2.4 |
| \$DEVN | 1252 | 17 | SEE SEC. 2.4 |
| SWREG | 176 | 0 | SEE SEC. 2.3 |

2.5 EXECUTION TIMES.

EXECUTION TIMES FOR AN LSI-11 PROCESSOR WITH ONE DLV11-J MODULE AT SHIPMENT CONFIGURATION:

CH. 0,1,2 AT 9600 BAUD.
 CH. 3 (CONSOLE) AT 300 BAUD.

ARE: FIRST PASS- 30 SEC
 ADDITIONAL PASSES 90 SEC

THE TEST TIME IS BAUD RATE DEPENDENT; HIGHER BAUD RATES RESULT IN SHORTER PASS TIMES.

2.6 POWER FAIL.

AUTO START FROM POWER FAIL IS NOT IMPLEMENTED IN THIS PROGRAM.

3.0 ERROR INFORMATION.

3.1 ERROR REPORTING PROCEDURE.

SINCE THIS DIAGNOSTIC WAS DESIGNED TO FIT IN 4K OF MEMORY THE ERROR TYPEOUT IS VERY BRIEF. THE FORMAT OF THE ERROR TYPEOUT IS AS FOLLOWS:

TEST#____,ERROR#____,PC=____,ADDRESS=____,VECTOR=____

WHERE ALL VALUES TYPED ARE OCTAL.
THE ADDRESS AND VECTOR REFER TO THE FAILING CHANNEL.
FOR FURTHER INFORMATION THE LISTING MUST BE CONSULTED.
BITS 15,13,10 AND 9 OF THE SWITCH REGISTER (SWREG) CONTROL THE SEQUENCE OF EVENTS AFTER AN ERROR IS CAUGHT.

BIT 15 SET: CAUSES THE PROGRAM TO HALT IN THE ERROR ROUTINE.
IF THE PROGRAM IS CONTINUED, IT WILL PROCEED FROM WHERE IT HALTED.

BIT 13 SET: DISABLES THE PRINTING OF THE ERROR MESSAGE.

BIT 10 SET: CAUSES THE BELL TO RING ON ERROR.

BIT 9 SET: CAUSES THE DIAGNOSTIC TO LOOP FROM BEGINNING OF TEST TO ERROR.

THE ERROR ROUTINE SUPPORTS THE CONTROL G <^G> FUNCTION.
REFER TO SECTION 2.3 FOR DETAILS.

3.2 ERROR HALTS.

THE ONLY HALT IN THIS DIAGNOSTIC IS IN THE ERROR ROUTINE, AND IS EXECUTED ONLY IF BIT 15 OF THE SWITCH REGISTER (SWREG) IS SET WHEN AN ERROR OCCURS.

4.0 PERFORMANCE AND PROGRESS REPORTS.

4.1 PERFORMANCE REPORTS. (BIT 12 SET IN THE SWITCH REGISTER 'SWREG')

AS EACH CHANNEL (4 CHANNELS/DLV11-J) COMPLETES ONE PASS OF THE DIAGNOSTIC THE FOLLOWING ITEMS ARE TYPED:

'CSR:-----' : THE BASE ADDRESS OF THE LINE UNDER TEST
'VECTOR:---' : THE ASSOCIATED VECTOR
'ERRORS:---' : THE TOTAL NUMBER OF ERRORS ON THIS DEVICE ON THIS PASS.

AFTER ALL MODULES & CHANNELS TO BE TESTED HAVE BEEN EXERCISED, AN END PASS STATEMENT IS TYPED:

'END PASS#-----.'

EXAMPLE OF PRINTOUT ASSUMING: 2 DLV11-J MODULES
ALL CHANNELS ENABLED
SEPERATE CONSOLE DEVICE
NO ERRORS

CNDLAAO DLV11-J TEST

SWR= 000000 NEW= 10000<CR>

(ENABLE PERFORMANCE REPORTS)

WILL TEST:

MODULE 1 CHANNEL 0 1 2 3
MODULE 2 CHANNEL 0 1 2 3

** PHASE 1 SUMMARY **

CSR: 176500, VECTOR: 000300, ERRORS: 0
CSR: 176510, VECTOR: 000310, ERRORS: 0
CSR: 176520, VECTOR: 000320, ERRORS: 0
CSR: 176530, VECTOR: 000330, ERRORS: 0

** PHASE 2 SUMMARY **

CSR: 176500, VECTOR: 000300, ERRORS: 0

** PHASE 1 SUMMARY **

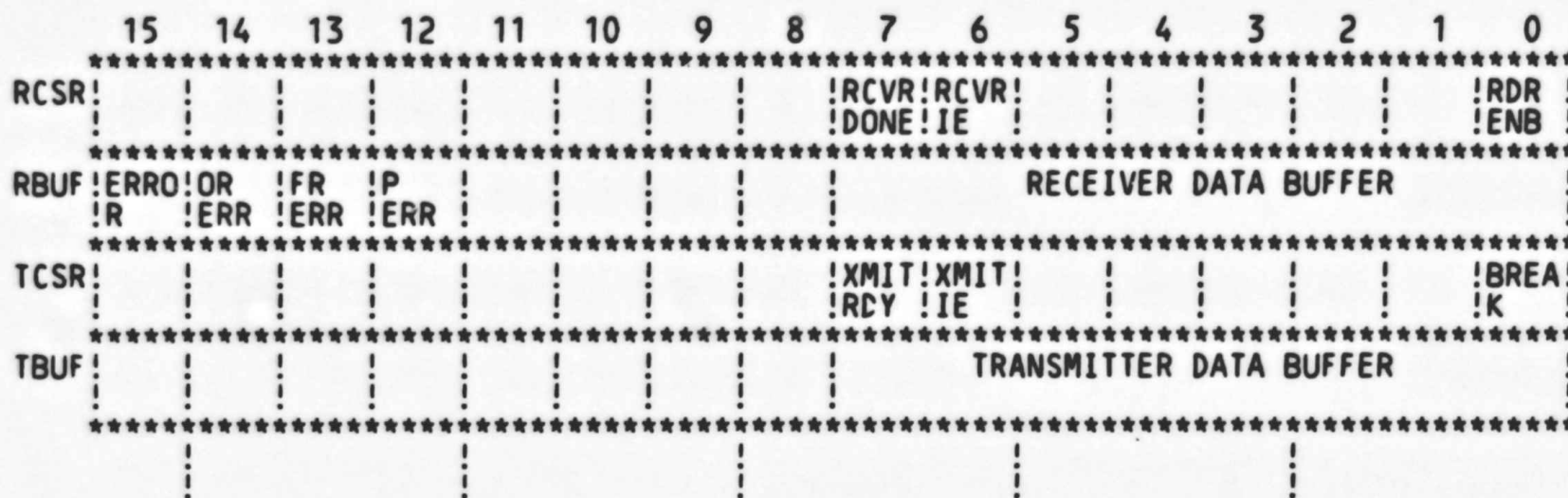
CSR: 176540, VECTOR: 000340, ERRORS: 0
CSR: 176550, VECTOR: 000350, ERRORS: 0
CSR: 176560, VECTOR: 000360, ERRORS: 0
CSR: 176570, VECTOR: 000370, ERRORS: 0

** PHASE 2 SUMMARY **

CSR: 176540, VECTOR: 000340, ERRORS: 0

END PASS # 1

5.0 DEVICE REGISTERS.



NOTES:

1. RCSR AT BASE ADDRESS (\$BASE)
 RBUF AT \$BASE+2
 TCSR AT \$BASE+4
 TBUF AT \$BASE+6
2. BLANK BOXES INDICATE UNUSED AND RESERVED BIT POSITIONS. SEE THE LISTING FOR AN EXPLANATION OF THE BITS.
3. ORERR = OVERRUN ERROR
 FRERR = FRAMING ERROR
 PERR = PARITY ERROR

6.0 SUMMARY OF TESTS AND SPECIAL SUBROUTINES.

PHASE 1 TESTS

TEST 1 ADDRESSABILITY

THIS TEST VERIFIES THAT ALL 4 REGISTERS OF THE CHANNEL
UNDER TEST RESPOND TO THEIR ADDRESSES.

THE FOLLOWING 3 TESTS TEST ALL 'READ WRITE' BITS

TEST 2 BREAK - TCSR 0 SET, CLEAR, RESET

TEST 3 XMITIE - TCSR 6 SET, CLEAR, RESET

TEST 4 RCVRIE - RCSR 6 SET, CLEAR, RESET

TEST 5 XMITRDY - TCSR 7 - IS SET BY INIT

TEST 6 XMIT RDY - TCSR 7 - CLEARS WHEN TBUF IS LOADED
----- WITH A CHARACTER AND THAT IT SETS WITHIN A
REASONABLE AMOUNT OF TIME.

TEST 7 OUTPUTTING A CHAR FROM TBUF (WITH WRAP AROUND CONNECTED)
----- RESULTS IN RCVRDONE SETTING WITHIN A
REASONABLE AMOUNT OF TIME AND THAT RESET
CLEARS THE BIT.

TEST 10 RCVRDONE IS CLEARED BY SETTING READER ENABLE

TEST 11 RCVRDONE IS CLEARED BY READING RBUF

TEST 12 OVERRUN & ERROR BIT - RBUF 14

TEST 13 TRANSMITTER INTERRUPT LOGIC TEST

----- --

LOGICALLY THIS IS 4 SEPARATE TESTS

- A) DOES TRANSMITTER INTERRUPT LOGIC WORK
- B) AT PRIORITY OF 0
- C) AND ONLY ONCE
- D) BUT NOT WITH INTERRUPT ENABLE CLEAR

TEST 14 RECEIVER INTERRUPT LOGIC TEST THIS TEST COVERS ALL
----- -- OF THE RECEIVER SIDE OF THE INTERRUPT LOGIC IN
CHARACTER MODE.

TEST 15 TEST DATA WRAP AROUND: FLAG MODE.

----- --

TEST 16 TEST DATA WRAP AROUND: INTERRUPT MODE.

----- --

TEST 17 TEST BREAK DETECTION LOGIC TRANSMIT KNOWN CHAR

----- --

- A) TRANSMIT KNOWN CHAR WITH BREAK SET
AND COMPARE RECEIVED WITH 0
- B) TEST FOR FRAMING ERROR ON BREAK
- C) IF PARITY IS ENABLED AND ODD PARITY IS SELECTED,
CHECK TO BE SURE PARITY ERROR WAS GENERATED
- D) IF PARITY IS ENABLED AND EVEN PARITY IS SELECTED,
CHECK TO BE SURE NO PARITY ERROR OCCURRED

TEST 20 NOT A TEST - SEND BACK TO LOOP

PHASE 2 TESTS

TEST 21 TEST THAT CHANNELS INTERRUPT AT ASSIGNED PRIORITY

----- --

TEST 22 TEST DATA TRANSFERS WITH ALL ACTIVE LINES INTERRUPTING.

----- --

%


```
(1) ;*PRIORITY LEVEL DEFINITIONS
(1) 000000 PR0= 0 ;:PRIORITY LEVEL 0
(1) 000040 PR1= 40 ;:PRIORITY LEVEL 1
(1) 000100 PR2= 100 ;:PRIORITY LEVEL 2
(1) 000140 PR3= 140 ;:PRIORITY LEVEL 3
(1) 000200 PR4= 200 ;:PRIORITY LEVEL 4
(1) 000240 PR5= 240 ;:PRIORITY LEVEL 5
(1) 000300 PR6= 300 ;:PRIORITY LEVEL 6
(1) 000340 PR7= 340 ;:PRIORITY LEVEL 7
```

```
(1) ;*'SWITCH REGISTER' SWITCH DEFINITIONS
(1) 100000 SW15= 100000
(1) 040000 SW14= 40000
(1) 020000 SW13= 20000
(1) 010000 SW12= 10000
(1) 004000 SW11= 4000
(1) 002000 SW10= 2000
(1) 001000 SW09= 1000
(1) 000400 SW08= 400
(1) 000200 SW07= 200
(1) 000100 SW06= 100
(1) 000040 SW05= 40
(1) 000020 SW04= 20
(1) 000010 SW03= 10
(1) 000004 SW02= 4
(1) 000002 SW01= 2
(1) 000001 SW00= 1
(1) .EQUIV SW09,SW9
(1) .EQUIV SW08,SW8
(1) .EQUIV SW07,SW7
(1) .EQUIV SW06,SW6
(1) .EQUIV SW05,SW5
(1) .EQUIV SW04,SW4
(1) .EQUIV SW03,SW3
(1) .EQUIV SW02,SW2
(1) .EQUIV SW01,SW1
(1) .EQUIV SW00,SW0
```

```
(1) ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1) 100000 BIT15= 100000
(1) 040000 BIT14= 40000
(1) 020000 BIT13= 20000
(1) 010000 BIT12= 10000
(1) 004000 BIT11= 4000
(1) 002000 BIT10= 2000
(1) 001000 BIT09= 1000
(1) 000400 BIT08= 400
(1) 000200 BIT07= 200
(1) 000100 BIT06= 100
(1) 000040 BIT05= 40
(1) 000020 BIT04= 20
(1) 000010 BIT03= 10
(1) 000004 BIT02= 4
(1) 000002 BIT01= 2
(1) 000001 BIT00= 1
(1) .EQUIV BIT09,BIT9
```



```

(1) .EQUIV BIT08,BIT8
(1) .EQUIV BIT07,BIT7
(1) .EQUIV BIT06,BIT6
(1) .EQUIV BIT05,BIT5
(1) .EQUIV BIT04,BIT4
(1) .EQUIV BIT03,BIT3
(1) .EQUIV BIT02,BIT2
(1) .EQUIV BIT01,BIT1
(1) .EQUIV BIT00,BIT0
(1)
(1) ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1) 000004 ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS
(1) 000010 RESVEC= 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
(1) 000014 TBITVEC=14 ;: "T" BIT
(1) 000014 TRTVEC= 14 ;:TRACE TRAP
(1) 000014 BPTVEC= 14 ;:BREAKPOINT TRAP (BPT)
(1) 000020 IOTVEC= 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1) 000024 PWRVEC= 24 ;:POWER FAIL
(1) 000030 EMTVEC= 30 ;:EMULATOR TRAP (EMT) **ERROR**
(1) 000034 TRAPVEC=34 ;: "TRAP" TRAP
(1) 000060 TKVEC= 60 ;:TTY KEYBOARD VECTOR
(1) 000064 TPVEC= 64 ;:TTY PRINTER VECTOR
(1) ;***** THE FOLLOWING BREAK VECTOR AND LINE CLOCK VECTOR ARE INCLUDED
(1) 000100 LKVEC= 100 ;:LINE CLOCK VECTOR
(1) 000140 BRKVEC= 140 ;:BREAK VECTOR
(1) 000240 PIRQVEC=240 ;:PROGRAM INTERRUPT REQUEST VECTOR

8725
8726
8727
8728 000001 ADRS= R1
8729 176500 DLADDR= 176500
8730
8731 ; THE FOLLOWING DEFINITIONS APPLY TO THE GLOBAL SUBS
8732 177777 SET= -1
8733 000000 CLR= 0
8734 000001 TRUE= 1
  
```

8736
8737
8738
8739
8740
8741
8742
8743
8744
8745
8746
8747
8748
8749
8750
8751
8752
8753
8754
8755
8756
8757
8758
8759
8760
8761
8762
8763
8764
8765
8766
8767
8768
8769
8770
8771
8772
8773
8774
8775
8776
8777

000200
000100

000001

100000
040000
020000
010000

000200
000100
000040
000020
000010
000004
000002
000001

```

:*****
: RCSR REGISTER BIT NAMES
:*****
: UNUSED BIT15
: UNUSED BIT14
: UNUSED BIT13
: UNUSED BIT12
: UNUSED BIT11
: UNUSED BIT10
: UNUSED BIT09
: UNUSED BIT08
RCVRDONE= BIT07 : RECEIVER DONE
RCVRIE= BIT06 : RECEIVER INTERRUPT ENABLE
: UNUSED BIT05
: UNUSED BIT04
: UNUSED BIT03
: UNUSED BIT02
: UNUSED BIT01
RDRRUN= BIT00 : READER RUN

:*****
: RBUF REGISTER BIT NAMES
:*****
ERROR= BIT15 : ERROR INDICATOR
ORERR= BIT14 : OVERRUN ERROR
FRERR= BIT13 : FRAMING ERROR
PERR= BIT12 : PARITY ERROR
: UNUSED BIT11
: UNUSED BIT10
: UNUSED BIT09
: UNUSED BIT08
RDATA7= BIT07
RDATA6= BIT06
RDATA5= BIT05
RDATA4= BIT04
RDATA3= BIT03
RDATA2= BIT02
RDATA1= BIT01
RDATA0= BIT00
  
```



8779
8780
8781
8782
8783
8784
8785
8786
8787
8788
8789
8790
8791
8792
8793
8794
8795
8796
8797
8798
8799
8800
8801
8802
8803
8804
8805
8806
8807
8808
8809
8810
8811
8812
8813
8814
8815
8816
8817
8818
8819
8820
8821
8822
8823
8824
8825
8826
8827
8828
8829

000200
000100

000001

000200
000100
000040
000020
000010
000004
000002
000001

000002
000004
000010
000020

 ; TCSR REGISTER BIT NAMES

```

: UNUSED BIT15
: UNUSED BIT14
: UNUSED BIT13
: UNUSED BIT12
: UNUSED BIT11
: UNUSED BIT10
: UNUSED BIT09
: UNUSED BIT08
XMITRDY= BIT07 ; TRANSMITTER READY
XMITIE= BIT06 ; TRANSMITTER INTERRUPT ENABLE
: UNUSED BIT05
: UNUSED BIT04
: UNUSED BIT03
: UNUSED BIT02
: UNUSED BIT01
BREAK= BIT00 ; SEND BREAK (CONTINUOUS SPACE)
  
```

 ; TBUF REGISTER BIT NAMES

```

: UNUSED BIT15
: UNUSED BIT14
: UNUSED BIT13
: UNUSED BIT12
: UNUSED BIT11
: UNUSED BIT10
: UNUSED BIT09
: UNUSED BIT08
TDATA7= BIT07 : \
TDATA6= BIT06 : |
TDATA5= BIT05 : |
TDATA4= BIT04 : |
TDATA3= BIT03 : |
TDATA2= BIT02 : |
TDATA1= BIT01 : |
TDATA0= BIT00 : /
          \ TRANSMITTER DATA BUFFER
          /
  
```

 ; FLAG BITS TO BE USE OR CLEARED IN \$USWR.

```

PARITY = 2
EVENODD = 4
BRK = 10
WRAP = 20
  
```

```

8831
8832
8833
8834
(1)
(1)          000000
(1)
(1)
(1)
(1)
(1)          000174
(1) 000174 000000
(1) 000176 000000
(1)
(1) 000200 000137 001334
8835
8836
8837
8838
8839
8840
8841          000400
8842
8843 000400 005000
8844 000402 012720 013610
8845 000406 012720 000300
8846
8847 000412 020027 000400
8848 000416 001371
8849 000420 005037 000176
8850 000424 000137 001334

;*****
;SBTTL TRAP CATCHER
      .=0
;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
      .=174
DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER
.SBTTL  STARTING ADDRESS(ES)
      JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM

;
; OFFLINE TESTING FOR INTERRUPT VECTOR PROBLEMS
;
      .=400
1$:   CLR    R0
      MOV    #INTSRV,(R0)+ ;SET INTR HANDLER PTR
      MOV    #PR6,(R0)+   ;SET PRIORITY TO 6 TO BE
;                                     ;SPECIFIC TO 11/21 PROCESSOR
      CMP    R0,#400      ;ALL DONE?
      BNE   1$           ;BR IF NO
      CLR   176          ;CLEAN UP SWREG
      JMP   START        ;GO DO IT
  
```

```

8852
8853
8854      .SBTTL  ACT11 HOOKS
(1)
(2)      ::*****
(1)      ::HOOKS REQUIRED BY ACT11
(1)      $SVPC=.          ;SAVE PC
(1)      .=46
(1) 000046 014450      $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(1)      000052      .=52
(1) 000052 000000      .WORD 0          ;;2)SET LOC.52 TO ZERO
(1)      000430      .=$SVPC          ;; RESTORE PC
8855      .=1000
8856      .SBTTL  APT PARAMETER BLOCK
(1)
(2)      ::*****
(1)      ::SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(2)      ::*****
(1)      .SX=.          ;;SAVE CURRENT LOCATION
(1)      .=24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(1) 000024 000200      200          ;;FOR APT START UP
(1)      000044      .=44          ;;POINT TO APT INDIRECT ADDRESS PNTR.
(1) 000044 001000      $APTHDR      ;;POINT TO APT HEADER BLOCK
(1)      001000      .=.$X          ;;RESET LOCATION COUNTER
(2)      ::*****
(1)      ::SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(1)      ::INTERFACE SPEC.
(1)
(1) 001000      $APTHD:
(1) 001000 000000      $HIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(1) 001002 001174      $MBADR: .WORD $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
(1) 001004 000031      $STM: .WORD 25.          ;;RUN TIM OF LONGEST TEST
(1) 001006 000144      $PASTM: .WORD 100.          ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(1) 001010 000144      $UNITM: .WORD 100.          ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(1) 001012 000030      .WORD SETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
  
```



```

(2) 001210 000000 $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
(2) 001212 000000 $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
(2) 001214          $ETABLE:          ;;APT ENVIRONMENT TABLE
(2) 001214          $ENV: .BYTE   AENV   ;;ENVIRONMENT BYTE
(2) 001215          $ENVM: .BYTE   AENVM  ;;ENVIRONMENT
(2)          ;;ENVIRONMENT MODE BITS
(2) 001216 000000 $$WREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
(2) 001220 000031 $USWR: .WORD   AUSWR   ;;USER SWITCHES
(2) 001222 000000 $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
(2)          ;;BITS 15-11=CPU TYPE
(2)          ;;11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)          ;;11/70=06,PDQ=07,Q=10
(2)          ;;BIT 10=REAL TIME CLOCK
(2)          ;;BIT 9=FLOATING POINT PROCESSOR
(2)          ;;BIT 8=MEMORY MANAGEMENT
(2) 001224          $MAMS1: .BYTE   AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
(2) 001225          $MTYP1: .BYTE   AMTYP1 ;;MEM. TYPE,BLK#1
(2)          ;;MEM. TYPE BYTE -- (HIGH BYTE)
(2)          ;;900 NSEC CORE=001
(2)          ;;300 NSEC BIPOLAR=002
(2)          ;;500 NSEC MOS=003
(2) 001226 000000 $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
(2)          ;;MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
(2) 001230          $MAMS2: .BYTE   AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
(2) 001231          $MTYP2: .BYTE   AMTYP2 ;;MEM. TYPE,BLK#2
(2) 001232 000000 $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
(2) 001234          $MAMS3: .BYTE   AMAMS3 ;;HIGH ADDRESS,M.S.BYTE
(2) 001235          $MTYP3: .BYTE   AMTYP3 ;;MEM. TYPE,BLK#3
(2) 001236 000000 $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
(2) 001240          $MAMS4: .BYTE   AMAMS4 ;;HIGH ADDRESS,M.S.BYTE
(2) 001241          $MTYP4: .BYTE   AMTYP4 ;;MEM. TYPE,BLK#4
(2) 001242 000000 $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
(2) 001244 000300 $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2) 001246 000000 $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2) 001250 176500 $BASE: .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2) 001252 000017 $DEV: .WORD   ADEV     ;;DEVICE MAP
(2) 001254          .MEXIT
    
```

(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)

.SBTTL ERROR POINTER TABLE

;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
;*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
;*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;* EM ::POINTS TO THE ERROR MESSAGE
;* DH ::POINTS TO THE DATA HEADER
;* DT ::POINTS TO THE DATA
;* DF ::POINTS TO THE DATA FORMAT

| | | |
|------|--------|--------|
| 8858 | 001254 | |
| 8859 | 001254 | 176500 |
| 8860 | 001256 | 000300 |
| 8861 | 001260 | 176500 |
| 8862 | 001262 | 176502 |
| 8863 | 001264 | 176504 |
| 8864 | 001266 | 176506 |
| 8865 | 001270 | 000000 |
| 8866 | 001272 | 000020 |
| 8867 | 001332 | 000000 |

\$ERRTB:
:: GLOBAL DATA
DLADD: DLADDR
DLVEC: 300
RCSR: DLADDR + 0
RBUF: DLADDR + 2
TCSR: DLADDR + 4
TBUF: DLADDR + 6
I: 0
.BLKW 20 ;FOR R5 STACK
R5STACK: .WORD 0


```

8870
8871 001334
(1)
(1)
(1) 001334 012706 001100
(1) 001340 005026
(1) 001342 022706 001140
(1) 001346 001374
(1) 001350 012706 001100
(1)
(1) 001354 012737 016210 000020
(1) 001362 012737 000300 000022
(1) 001370 012737 016010 000030
(1) 001376 012737 000300 000032
(1)
(1) 001404 012737 017142 000034
(1) 001412 012737 000300 000036
(1) 001420 013737 014416 014410
(1) 001426 005037 001160
(1) 001432 005037 001162
(1) 001436 112737 000001 001115
(1) 001444 012737 001444 001106
(1) 001452 012737 001452 001110
(2)
(2)
(2) 001460 013746 000004
(2) 001464 012737 001520 000004
(2) 001472 012737 177570 001140
(2) 001500 012737 177570 001142
(2) 001506 022777 177777 177424
(2) 001514 001012
(2)
(2) 001516 000403
(2) 001520 012716 001526 64$:
(2) 001524 000002
(2) 001526 012737 000176 001140 65$:
(2) 001534 012737 000174 001142
(2) 001542 012637 000004 66$:
(1)
(2) 001546 005037 001202
(2) 001552 132737 000200 001215
(2) 001560 001403
(2) 001562 012737 001216 001140
(2) 001570
8872
(1)
(1) 001570 005227 177777
(1) 001574 001043
(1) 001576 022737 014450 000042
(1) 001604 001437
(1) 001606 104401 001654
(2)
(2) 001612 005737 000042
(2) 001616 001012
(2) 001620 123727 001214 000001
(2) 001626 001406
  
```

```

START:
.SBTTL INITIALIZE THE COMMON TAGS
::CLEAR THE COMMON TAGS ($CMTAG) AREA
MOV # $CMTAG,R6 ::FIRST LOCATION TO BE CLEARED
CLR (R6)+ ::CLEAR MEMORY LOCATION
CMP #SWR,R6 ::DONE?
BNE #-6 ::LOOP BACK IF NO
MOV #STACK,SP ::SETUP THE STACK POINTER
::INITIALIZE A FEW VECTORS
MOV # $SCOPE,@#IOTVEC ::IOT VECTOR FOR SCOPE ROUTINE
MOV #PR6,@#IOTVEC+2 ::LEVEL 6
MOV # $ERROR,@#EMTVEC ::EMT VECTOR FOR ERROR ROUTINE
MOV #PR6,@#EMTVEC+2 ::LEVEL 6
::BIT02
MOV #STRAP,@#TRAPVEC ::TRAP VECTOR FOR TRAP CALLS
MOV #PR6,@#TRAPVEC+2::LEVEL 6
MOV $ENDCT,$EOPCT ::SETUP END-OF-PROGRAM COUNTER
CLR $TIMES ::INITIALIZE NUMBER OF ITERATIONS
CLR $ESCAPE ::CLEAR THE ESCAPE ON ERROR ADDRESS
MOVB #1,$ERMAX ::ALLOW ONE ERROR PER TEST
MOV #,$LPADR ::INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV #,$LPERR ::SETUP THE ERROR LOOP ADDRESS
::SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
::EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV @#ERRVEC,-(SP) ::SAVE ERROR VECTOR
MOV #64,$@#ERRVEC ::SET UP ERROR VECTOR
MOV #DSWR,SWR ::SETUP FOR A HARDWARE SWICH REGISTER
MOV #DDISP,DISPLAY ::AND A HARDWARE DISPLAY REGISTER
CMP #-1,@SWR ::TRY TO REFERENCE HARDWARE SWR
BNE 66$ ::BRANCH IF NO TIMEOUT TRAP OCCURRED
::AND THE HARDWARE SWR IS NOT = -1
BR 65$ ::BRANCH IF NO TIMEOUT
MOV #65$,(SP) ::SET UP FOR TRAP RETURN
MOV #SWREG,SWR ::POINT TO SOFTWARE SWR
MOV #DISPREG,DISPLAY
MOV (SP)+,@#ERRVEC ::RESTORE ERROR VECTOR
CLR $PASS ::CLEAR PASS COUNT
BITB #APTSIZE,$ENVM ::TEST USER SIZE UNDER APT
BEQ 67$ ::YES,USE NON-APT SWITCH
MOV #SSWREG,SWR ::NO,USE APT SWITCH REGISTER
67$:
.SBTTL TYPE PROGRAM NAME
::TYPE THE NAME OF THE PROGRAM IF FIRST PASS
INC #-1 ::FIRST TIME?
BNE 68$ ::BRANCH IF NO
CMP # $ENDAD,@#42 ::ACT-11?
BEQ 68$ ::BRANCH IF YES
TYPE ,69$ ::TYPE ASCIZ STRING
.SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
TST @#42 ::ARE WE RUNNING UNDER XXDP/ACT?
BNE 70$ ::BRANCH IF YES
CMPB $ENV,#1 ::ARE WE RUNNING UNDER APT?
BEQ 70$ ::BRANCH IF YES
  
```

```
(2) 001630 023727 001140 000176      CMP      SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
(2) 001636 001005                      BNE      71$            ;;BRANCH IF NO
(2) 001640 104406                      GTSWR                      ;;GET SOFT-SWR SETTINGS
(2) 001642 000403                      BR       71$
(2) 001644 112737 000001 001134 70$:  MOVB     #1,$AUTOB      ;;SET AUTO-MODE INDICATOR
(2) 001652 71$:
(1) 001652 000414                      BR       68$            ;;GET OVER THE ASCIZ
(1)          ;;69$: .ASCIZ <CRLF>*CNDLAAO DLV11-J TEST*<CRLF>
(1) 001704 68$:
```

```
8874
8875
8876
8877 001704 004737 012700          JSR    PC,SIZE          ;SIZE THE BUS IF $DEVN = 0
8878
8879 001710 005037 014166          CLR    PHASE2          ;INIT FOR TESTS
8880 001714 012737 000001 001206    MOV    #1,$UNIT
8881 001722 005037 014164          CLR    CHAN
8882 001726 005037 014170          CLR    P1CNT
8883
8884 001732 012706 001100          LOOP: MOV    #STACK,SP  ;RESET STACK POINTER
8885 001736 004737 013620          JSR    PC,CYCLE       ;SETUP MODULE AND CHANNEL
8886
8887
8888
8889
8890
8891
8892 001742
(4) 001742 012137 001254          MOV    (ADRS)+,DLADD  ;GET UNIT ADDRESS
8893
8894 001746
(4) 001746 011137 001256          MOV    (ADRS),DLVEC  ;GET UNIT VECTOR
8895
8896 001752
(4) 001752 013737 001254 001260    MOV    DLADD,RCSR    ;RCSR = DLADD + 0
8897 001760
(4) 001760 013737 001254 001262    MOV    DLADD,RBUF    RCSR := DLADD
8898 001774
(4) 001774 013737 001254 001264    MOV    DLADD,TCSR    RBUF := DLADD + #2
8899 002010
(4) 002010 013737 001254 001266    MOV    DLADD,TBUF    TCSR := DLADD + #4
8900 002024
(4) 002024 012705 001332          MOV    #R5STACK,R5  TBUF := DLADD + #6
8901
8902 002030 012700 001000          MOV    #1000,R0     R5 := #R5STACK
8903 002034 077001
8904 002036
(1) 002036 000005          RESET                ;DELAY TO ALLOW PRINT TO FINISH
8905 002040
(7) 002040 005237 001204          INC    $DEVCT       ;BEFORE RESET
8906 002044
(6) 002044 123727 014166 000001    CMPB  PHASE2,#TRUE  BRESET
8907 002054 000137 007000          JMP    MODTST
8908 002060
(4) 002060 000402          BR     $2            ;GO TO PHASE 2 TESTING
8909 002062
(4) 002062 005037 001102          CLR    $STNM        ELSE
8910 002066
(4) 002066          $1:
$2:
      LET $STNM := #0
      ENDIF
```



```
8922  
8923  
(3)  
(4)  
(4)  
(4)  
(4)  
(3)  
(2) 002066 000004  
(2)  
(1) 002070 012737 000002 001160  
(2) 002076 012737 000001 001200  
8928 002104  
(4) 002104 013701 001254  
8929  
8930 002110 012737 013610 000004  
8931 002116 012737 000300 000006  
8932  
8933 002124  
(4) 002124 005037 001270  
8934 002130  
(3) 002130  
8935 002130  
(5) 002130 012737 002136 001110  
8936  
8937 002136  
(4) 002136 005037 013616  
8938  
8939  
8940 002142 005711  
8941 002144  
(6) 002144 005737 013616  
(9) 002150 001401  
8942  
8943 002152  
(1) 002152 104001  
8944 002154  
(4) 002154  
8945 002154  
8946 002154  
(7) 002154 062737 000002 001270  
8947 002162  
(4) 002162 013701 001254  
(7) 002166 063701 001270  
8948 002172  
(3) 002172 023727 001270 000010  
(6) 002200 001353
```

```
*****  
*TEST 1 ADDRESSABILITY  
* THIS TEST VERIFIES THAT ALL 4 REGISTERS OF THE CHANNEL UNDER TEST  
* RESPOND TO THEIR ADDRESSES.  
*****  
TST1: SCOPE  
MOV #2,$TIMES ;;DO 2 ITERATIONS  
MOV #1,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
LET ADRS := DLADD  
; SET UP INTERRUPT  
MOV #INTSRV,ERRVEC  
MOV #PR6,ERRVEC+2 ;PRIORITY LOWERED TO 6  
;TO BE SPECIFIC TO PDP 11/21  
LET I := #0  
REPEAT  
BGNSUB  
;CLEAR FLAG  
LET INTFLAG := #0  
;READ FLAG  
IF INTFLAG NE #0 THEN  
; FATAL ERROR  
ERRDF 1  
ENDIF  
ENDSUB  
LET I := I + #2  
LET ADRS := DLADD + I  
UNTIL I EQ #8.  
$3:  
MOV #64,$,SLPERR  
CLR INTFLAG  
TST @ADRS  
TST INTFLAG  
BEQ $4  
ERROR 1  
$4:
```

| | | | | | | | | |
|------|--------|--------|--------|--------|------|-------|---------------|-----------------------------------|
| 8950 | | | | | | | | |
| 8951 | | | | | | | | |
| 8952 | 002202 | 032777 | 177476 | 177050 | | BIT | #177476,@RCSR | ;CHECK THAT ALL UNUSED BITS ARE 0 |
| 8953 | 002210 | 001401 | | | | BEQ | 1\$ | |
| 8954 | 002212 | 104027 | | | | ERROR | 27 | ;RCSR HAS UNUSED BITS SET |
| 8955 | | | | | | | | |
| 8956 | 002214 | 032777 | 007400 | 177040 | 1\$: | BIT | #7400,@RBUF | |
| 8957 | 002222 | 001401 | | | | BEQ | 2\$ | |
| 8958 | 002224 | 104030 | | | | ERROR | 30 | ;RBUF HAS UNUSED BITS SET |
| 8959 | | | | | | | | |
| 8960 | 002226 | 032777 | 177476 | 177030 | 2\$: | BIT | #177476,@TCSR | |
| 8961 | 002234 | 001401 | | | | BEQ | 3\$ | |
| 8962 | 002236 | 104031 | | | | ERROR | 31 | ;TCSR HAS UNUSED BITS SET |
| 8963 | | | | | | | | |
| 8964 | 002240 | 032777 | 177400 | 177020 | 3\$: | BIT | #177400,@TBUF | |
| 8965 | 002246 | 001401 | | | | BEQ | 4\$ | |
| 8966 | 002250 | 104032 | | | | ERROR | 32 | ;TBUF HAS ;UNUSED BITS SET |
| 8967 | | | | | | | | |
| 8968 | 002252 | 012737 | 000006 | 000004 | 4\$: | MOV | #6,ERRVEC | ;RESTORE |
| 8969 | 002260 | 005037 | 000006 | | | CLR | ERRVEC+2 | |
| 8970 | | | | | | | | |

8972
8977
8978
8979
8980
8981
8986
(3)
(4)
(4)
(3)
(2) 002264 000004
(2)
(1) 002266 012737 000010 001160
(2) 002274 012737 000002 001200
8987
8992 002302
(6) 002302 032737 000010 001220
(9) 002310 001001
8993 002312
(3) 002312 000452
8994 002314
(4) 002314
8995
8996
8997 002314
(5) 002314 012737 002322 001110
8998
8999 002322
(6) 002322 032777 000001 176734
(9) 002330 001401
9000
9001 002332
(1) 002332 104002
9002 002334
(4) 002334
9003 002334
9004
9005
9006 002334
(5) 002334 012737 002342 001110
9007 002342
(7) 002342 052777 000001 176714
9008
9009 002350
(6) 002350 032777 000001 176706
(9) 002356 001001
9010
9011 002360
(1) 002360 104003
9012 002362
(4) 002362
9013 002362
9014
9015
9016 002362

* THE FOLLOWING 3 TESTS TEST ALL 'READ WRITE' BITS

*TEST 2 BREAK - TCSR 0 SET, CLEAR, RESET
* THIS BIT IS THE ONLY ONE IN THIS POSITION
* THAT IS READ AND WRITE.

TST2: SCOPE

MOV #10,\$TIMES ;:DO 10 ITERATIONS
MOV #2,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX

IF #BRK NOTSETIN \$USWR THEN

BIT #BRK,\$USWR
BNE \$5

BR TST3 ;:EXIT THIS TEST
ENDIF

; BREAK NOT INSTALLED

\$5:

; SEE IF IT IS CLEAR
BGNSUB

MOV #64,\$LPERR

IF #BREAK SETIN @TCSR THEN

BIT #BREAK,@TCSR
BEQ \$6

; BREAK DID NOT RESET IN TCSR
ERRHRD 2

ERROR 2

ENDIF

\$6:

ENDSUB

; TRY TO SET BREAK BIT
BGNSUB

MOV #64,\$LPERR

LET @TCSR := @TCSR SET.BY #BREAK

BIS #BREAK,@TCSR

; STUCK TO 0
IF #BREAK NOTSETIN @TCSR THEN

BIT #BREAK,@TCSR
BNE \$7

; BREAK DID NOT SET IN TCSR
ERRHRD 3

ERROR 3

ENDIF

\$7:

ENDSUB

; TRY TO CLEAR A SET BIT
BGNSUB


```

(5) 002362 012737 002370 001110      MOV      #64$, $LPERR
9017
9018 002370                                LET      @TCSR := @TCSR CLR.BY #BREAK
(7) 002370 042777 000001 176666      BIC      #BREAK, @TCSR
9019                                     ; SHOULD HAVE CLEARED
9020 002376                                IF      #BREAK SETIN @TCSR THEN
(6) 002376 032777 000001 176660      BIT      #BREAK, @TCSR
(9) 002404 001401                        BEQ      $10
9021                                     ; BREAK DID NOT CLEAR IN TCSR
9022 002406                                ERRHRD 4
(1) 002406 104004                        ERROR   4
9023 002410                                ENDIF
(4) 002410                                $10:
9024 002410                                ENDSUB
9025
9026                                     ; NOW SEE IF RESET CLEARS IT
9027 002410                                BGNSUB
(5) 002410 012737 002416 001110      MOV      #64$, $LPERR
9028
9029 002416                                LET      @TCSR := @TCSR SET.BY #BREAK
(7) 002416 052777 000001 176640      BIS      #BREAK, @TCSR
9030                                     ; ISSUE BUS RESET
9031 002424                                BRESËT
(1) 002424 000005                        RESET
9032 002426                                IF      #BREAK SETIN @TCSR THEN
(6) 002426 032777 000001 176630      BIT      #BREAK, @TCSR
(9) 002434 001401                        BEQ      $11
9033                                     ; BREAK DID NOT RESET IN TCSR
9034 002436                                ERRHRD 5
(1) 002436 104005                        ERROR   5
9035 002440                                ENDIF
(4) 002440                                $11:
9036 002440                                ENDSUB
  
```

```

9038
9039
9044      ;*****
(3)      ;*TEST 3          XMITIE - TCSR 6          SET, CLEAR, RESET
(3)      ;*****
(2) 002440 000004      TST3: SCOPE
(2)
(1) 002442 012737 000010 001160      MOV #10,$TIMES      ;;DO 10 ITERATIONS
(2) 002450 012737 000003 001200      MOV #3,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9045                                     ; USE PRIORITY OF 6
9050 002456      SETPRI #PR6
9051
9052                                     ; SEE IF IT IS CLEAR
9053 002470                                     BGNSUB
(5) 002470 012737 002476 001110      MOV #65$,$LPERR
9054
9055 002476                                     IF #XMITIE SETIN @TCSR THEN
(6) 002476 032777 000100 176560      BIT #XMITIE,@TCSR
(9) 002504 001401      BEQ $12
9056                                     ; XMITIE DID NOT RESET IN TCSR
9057 002506                                     ERRHRD 12
(1) 002506 104012      ERROR 12
9058 002510
(4) 002510      $12:
9059 002510                                     ENDSUB
9060
9061                                     ; TRY TO SET XMITIE BIT
9062 002510                                     BGNSUB
(5) 002510 012737 002516 001110      MOV #64$,$LPERR
9063 002516                                     LET @TCSR := @TCSR SET.BY #XMITIE
(7) 002516 052777 000100 176540      BIS #XMITIE,@TCSR
9064                                     ; STUCK TO 0
9065 002524                                     IF #XMITIE NOTSETIN @TCSR THEN
(6) 002524 032777 000100 176532      BIT #XMITIE,@TCSR
(9) 002532 001001      BNE $13
9066                                     ; XMIT DID NOT SET IN TCSR
9067 002534                                     ERRHRD 13
(1) 002534 104013      ERROR 13
9068 002536
(4) 002536      $13:
9069 002536                                     ENDSUB
9070
9071                                     ; TRY TO CLEAR A SET BIT
9072 002536                                     BGNSUB
(5) 002536 012737 002544 001110      MOV #64$,$LPERR
9073
9074 002544                                     LET @TCSR := @TCSR CLR.BY #XMITIE
(7) 002544 042777 000100 176512      BIC #XMITIE,@TCSR
9075                                     ; SHOULD HAVE CLEARED
9076 002552                                     IF #XMITIE SETIN @TCSR THEN
(6) 002552 032777 000100 176504      BIT #XMITIE,@TCSR
(9) 002560 001401      BEQ $14
9077                                     ; XMIT DID NOT CLEAR IN TCSR
9078 002562                                     ERRHRD 14
(1) 002562 104014      ERROR 14
9079 002564                                     ENDIF
  
```

```

(4) 002564
9080 002564
9081
9082
9083 002564 012737 002572 001110
(5) 002564 012737 002572 001110 MOV #64$, $LPERR
9084
9085 002572 052777 000100 176464 BIS #XMITIE, @TCSR LET @TCSR := @TCSR SET.BY #XMITIE
(7) 002572 052777 000100 176464
9086
9087 002600 000005 RESET ; ISSUE BUS RESET
(1) 002600 000005
9088 002602 032777 000100 176454 BIT #XMITIE, @TCSR BRESËT
(6) 002602 032777 000100 176454 BEQ $15 IF #XMITIE SET IN @TCSR THEN
(9) 002610 001401
9089
9090 002612 104015 ERROR 15 ; XMIT DID NOT RESET IN TCSR
(1) 002612 104015 ERRHRD 15
9091 002614
(4) 002614
9092 002614
  
```

\$14:

ENDSUB

; NOW SEE IF RESET CLEARS IT
 BGNSUB

\$15:

ENDIF

ENDSUB


```

9094
9095
9104
(3)
(4)
(4)
(3)
(2) 002614 000004
(2)
(1) 002616 012737 000010 001160
(2) 002624 012737 000004 001200
9109
9110 002632
(5) 002632 012737 002640 001110
9111
9112 002640
(6) 002640 032777 000100 176412
(9) 002646 001401
9113
9114 002650
(1) 002650 104035
9115 002652
(4) 002652
9116 002652
9117
9118
9119 002652
(5) 002652 012737 002660 001110
9120 002660
(7) 002660 052777 000100 176372
9121
9122 002666
(6) 002666 032777 000100 176364
(9) 002674 001001
9123
9124 002676
(1) 002676 104036
9125 002700
(4) 002700
9126 002700
9127
9128
9129 002700
(5) 002700 012737 002706 001110
9130
9131 002706
(7) 002706 042777 000100 176344
9132
9133 002714
(6) 002714 032777 000100 176336
(9) 002722 001401
9134
9135 002724
(1) 002724 104037
9136 002726
(4) 002726

```

```

*****
:TEST 4          RCVRIE - RCSR 6          SET, CLEAR, RESET
:                THIS BIT IS THE ONLY ONE IN THIS POSITION
:                THAT IS READ AND WRITE.
*****
TST4:  SCOPE
      MOV #10,$TIMES          ;;DO 10 ITERATIONS
      MOV #4,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
                          ; SEE IF IT IS CLEAR
                          BGNSUB
      MOV #64,$LPERR
      IF #RCVRIE SETIN @RCSR THEN
      BIT #RCVRIE,@RCSR
      BEQ $16
                          ; RCVRIE DID NOT RESET IN RCSR
                          ERRHRD 35
      ENDIF
      ENDSUB
                          ; TRY TO SET RCVRIE BIT
                          BGNSUB
      MOV #64,$LPERR
      BIS #RCVRIE,@RCSR
      LET @RCSR := @RCSR SET.BY #RCVRIE
      ; STUCK TO 0
      IF #RCVRIE NOTSETIN @RCSR THEN
      ; RCVRIE DID NOT SET IN RCSR
      ERRHRD 36
      ENDIF
      ENDSUB
                          ; TRY TO CLEAR A SET BIT
                          BGNSUB
      MOV #64,$LPERR
      BIC #RCVRIE,@RCSR
      LET @RCSR := @RCSR CLR.BY #RCVRIE
      ; SHOULD HAVE CLEARED
      IF #RCVRIE SETIN @RCSR THEN
      BIT #RCVRIE,@RCSR
      BEQ $20
                          ; RCVRIE DID NOT CLEAR IN RCSR
                          ERRHRD 37
      ENDIF

```

```

$16:
$17:
$20:

```

```

9137 002726                                ENDSUB
9138
9139                                ; NOW SEE IF RESET CLEARS IT
9140 002726                                BGNSUB
(5) 002726 012737 002734 001110          MOV    #64$, $LPERR
9141
9142 002734                                LET   @RCSR := @RCSR SET.BY #RCVRIE
(7) 002734 052777 000100 176316          BIS   #RCVRIE, @RCSR
9143
9144 002742                                ; ISSUE BUS RESET
(1) 002742 000005                          BRESËT
9145 002744                                IF   #RCVRIE SETIN @RCSR THEN
(6) 002744 032777 000100 176306          BIT   #RCVRIE, @RCSR
(9) 002752 001401                          BEQ   $21
9146
9147 002754                                ; RCVRIE DID NOT RESET IN RCSR
(1) 002754 104040                          ERRHRD 40
9148 002756                                ENDIF
(4) 002756
9149 002756                                ENDSUB
$21:
  
```

```

9151
9152
9157      ;*****
          ;*TEST 5      TEST THAT XMITRDY - TCSR 7 - IS SET BY INIT
          ;*****
          TST5:  SCOPE
          (2) 002756 000004
          (2)
          (1) 002760 012737 000010 001160      MOV #10,$TIMES      ;;DO 10 ITERATIONS
          (2) 002766 012737 000005 001200      MOV #5,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
9158
9163 002774
          (5) 002774 012737 003002 001110      MOV #64$,$LPERR      BGNSUB
9164
9165 003002
          (6) 003002 032777 000200 176254      BIT #XMITRDY,@TCSR      IF #XMITRDY NOTSETIN @TCSR THEN
          (9) 003010 001002      BNE $22
9166
9167
9168
9169 003012
          (1) 003012 104042      ERROR 42      ;RESET SHOULD HAVE SET BIT.
          ;XMITRDY DID NOT SET IN TCSR (AFTER RESE
          ERRHRD 42
9170
9171 003014
          (1) 003014 000005      RESET      ;ISSUE ANOTHER RESET
          BRESET
9172 003016
          (4) 003016      $22:      ENDIF
9173
9174 003016      ENDSUB      ;ALLOW LOOPING ON ERROR
  
```



```

9176
9177
9186
(3)
(4)
(4)
(3)
(2) 003016 000004
(2)
(1) 003020 012737 000010 001160 MOV #10,$TIMES ;;DO 10 ITERATIONS
(2) 003026 012737 000006 001200 MOV #6,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9191
9192 003034 IF CONSOL EQ #TRUE THEN
(6) 003034 023727 013604 000001 CMP CONSOL,#TRUE
(9) 003042 001001 BNE $23
9193 003044
(3) 003044 000521 BR TST7 ;;EXIT THIS TEST
9194 003046
(4) 003046 $23: EXIT
ENDIF
9195
9196 003046 LET PASS := #0 ;INIT COUNT OF TIMES THRU
(4) 003046 005037 003302 CLR PASS LOOP ; START OF LOOP
9197 003052
(3) 003052 $24: ; MAX OF 2 TIMES THRU
9198
9199 003052 LET ERRORFLAG := #0
(4) 003052 005037 003304 CLR ERRORFLAG LET EXITFLAG := #0
9200 003056
(4) 003056 005037 003306 CLR EXITFLAG
9201 : LOAD TBUF WITH ONE CHARACTER
9202 : WAIT FOR READY TO SET
9203 : (SHOULD BE VERY SHORT WAIT
9204 : SINCE UART DOUBLE BUFFERS ITS INPUT)
9205
9206 :SEND A CHARACTER
9207 003062 LET @TBUF :B= #0
(4) 003062 105077 176200 CLRB @TBUF
9208
9209 :WAIT A MAXIMUM
9210 :OF 50 MSEC FOR
9211 :XMIT RDY TO SET IN TCSR
(3) 003066 CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>
(7) 003070 MOV R5,-(SP)
(6) 003074 MOV #SET,-(R5)
(5) 003100 MOV TCSR,-(R5)
(4) 003104 MOV #XMITRDY,-(R5)
(3) 003110 MOV #500,-(R5)
(3) 003114 JSR PC,TIMER
MOV (SP)+,R5
9212
9213 :TIMER RETURNS AN ERROR IF BIT DID
9214 003116 :NOT MEET CONDITION WITHIN TIME LIMIT
(6) 003116 103003 BCC $26 IF.ERROR THEN
9215
9216 003120 :XMIT RDY DID NOT SET IN TCSR
(1) 003120 104066 ERROR 66
9217 003122 000137 003310 JMP TST7 ERRHRD 66
    
```

```

9218 003126                                     ENDIF
    (4) 003126
9219 003126 105077 176134
9220 003132 105777 176126
9221 003136 100375
9222
9223
9224
9225
9226
9227 003140                                     ;SEND SECOND CHARACTER
    (4) 003140 105077 176122                CLRB @TBUF :LET @TBUF :B= #0
9228 003144 000240
9229
9230
9231 003146                                     ; GIVE IT TIME TO CLEAR
    (6) 003146 032777 000200 176110        BIT #XMITRDY,@TCSR
    (9) 003154 001404                       BEQ $27
9232
9233
9234 003156                                     ; XMITRDY DID NOT CLEAR IN TCSR
    (4) 003156 012737 177777 003304        MOV #SET,ERRORFLAG ; WILL RESULT IN ERR 67 IF FAILS 2X
9235
9236
9237 003164                                     ; DEFER ERROR TYPEOUT
    (4) 003164 000416                       BR $30 ELSE
    (3) 003166
9238
9239
9240
9241 003166                                     ;WAIT A MAXIMUM
    (3) 003166 010546                       MOV R5,-(SP) ;OF 100 MSEC FOR
    (7) 003170 012745 177777                MOV #SET,-(R5) ;XMIT RDY TO SET IN TCSR
    (6) 003174 013745 001264                MOV TCSR,-(R5) CALL TIMER IN <#1000,#XMITRDY,TCSR,#SET>
    (5) 003200 012745 000200                MOV #XMITRDY,-(R5)
    (4) 003204 012745 001000                MOV #1000,-(R5)
    (3) 003210 004737 012446                JSR PC,TIMER
    (3) 003214 012605                MOV (SP)+,R5
9242 003216
    (6) 003216 103001                       BCC $31
9243
9244 003220
    (1) 003220 104070                       ERROR 70
9245 003222
    (4) 003222
9246 003222
    (4) 003222
9247 003222
    (6) 003222 023727 003304 177777        CMP ERRORFLAG,#SET
    (9) 003230 001013                       BNE $32
9248 003232
    (7) 003232 005237 003302                INC PASS
9249 003236
    (6) 003236 023727 003302 000001        CMP PASS,#1
    (9) 003244 003404                       BLE $33
9250
    ; CALL ERROR IF 2ND TRY
    
```

```

9251
9252 003246
(1) 003246 104067 ERROR 67
9253 003250
(4) 003250 012737 177777 003306 MOV #SET,EXITFLAG
9254 003256
(4) 003256 $33:
9255 003256
(4) 003256 000403 BR $34
(3) 003260 $32:
9256 003260
(4) 003260 012737 177777 003306 MOV #SET,EXITFLAG
9257 003266
(4) 003266 $34:
9258 003266
(3) 003266 023727 003306 177777 CMP EXITFLAG,#SET
(5) 003274 001401 BEQ $25
9259 003276
(4) 003276 000665 BR $24
(3) 003300 $25:
9260 003300
(3) 003300 000403 BR TST7
9261 003302 000000
9262 003304 000000
9263 003306 000000
  
```

: ON XMIT RDY NOT CLEARING
 ERRHRD 67

LET EXITFLAG := #SET

ENDIF

ELSE ; NO ERROR

LET EXITFLAG := #SET

ENDIF

EXIF EXITFLAG EQ #SET

ENDLOOP

EXIT : SKIP AROUND FLAG WORDS
 ;;EXIT THIS TEST

PASS: 0
 ERRORFLAG: 0
 EXITFLAG: 0


```
9265
9266
9275
(3)
(4)
(4)
(3)
(2) 003310 000004
(2)
(1) 003312 012737 000010 001160
(2) 003320 012737 000007 001200
9276
9281 003326
(6) 003326 032737 000020 001220
(8) 003334 001404
(6) 003336 023727 013604 000001
(9) 003344 001001
(6) 003346
9282 003346
(3) 003346 000434
9283 003350
(4) 003350
9284
9285 003350
(5) 003350 012737 003356 001110
9286
9287
9288 003356
(4) 003356 105077 175704
9289
9290
9291
9292
9293 003362
(3) 003362 010546
(7) 003364 012745 177777
(6) 003370 013745 001260
(5) 003374 012745 000200
(4) 003400 012745 000500
(3) 003404 004737 012446
(3) 003410 012605
9294
9295
9296 003412
(6) 003412 103001
9297
9298 003414
(1) 003414 104071
9299 003416
(4) 003416
9300
9301 003416
9302
9303 003416
(5) 003416 012737 003424 001110
9304
```

```
*****
:TEST 7 TEST THAT OUTPUTTING A CHAR FROM TBUF (WITH WRAP CONNECTED)
:* RESULTS IN RCVRDONE SETTING WITHIN A REASONABLE AMOUNT OF TIME
:* AND THAT RESET CLEARS THE BIT.
*****
TST7: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #7,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
BIT #WRAP,$USWR
BEQ $35
CMP CONSOL,#TRUE
BNE $36
$35: BR TST10 ;;EXIT THIS TEST
EXIT
ENDIF
BGNSUB
MOV #64,$LPERR ; SEND A CHARACTER AND LET IT WRAP AROUND
CLR B @TBUF LET @TBUF :B= #0
; WAIT A MAXIMUM OF 50 MSEC
; FOR RCVR DONE TO SET IN
; RCSR
CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
MOV R5,-(SP)
MOV #SET,-(R5)
MOV RCSR,-(R5)
MOV #RCVRDONE,-(R5)
MOV #500,-(R5)
JSR PC,TIMER
MOV (SP)+,R5
;DIDN'T SET IN TIME
IF.ERROR THEN
; RCVRDONE DID NOT SET IN RCSR
ERRHRD 71
ENDIF
$37: ENDSUB
BGNSUB
MOV #64,$LPERR ; NOW THAT IT IS SET SEE IF IT CAN BE RESET
```

```
9305 003424 BRESET  
(1) 003424 000005 RESET  
9306  
9307 003426 IF #RCVRDONE SETIN @RCSR THEN  
(6) 003426 032777 000200 175624 BIT #RCVRDONE,@RCSR  
(9) 003434 001401 BEQ $40  
9308 ; RCVRDONE DID NOT RESET IN RCSR.  
9309 003436 ERRHRD 72  
(1) 003436 104072 ERROR 72  
9310 003440 ENDIF  
(4) 003440 $40:  
9311 003440 ENDSUB
```


(6) 003554 032777 000200 175476
(9) 003562 001401
9351
9352 003564
(1) 003564 104026
9353
9354 003566
(4) 003566

BIT #RCVRDONE,@RCSR
BEQ \$44

ERROR 26

\$44:

:RCVRDONE DID NOT CLEAR IN RCSR
ERRHRD 26
: SET IT BACK TO CONTINUE
ENDIF

```

9356
9357
9362 *****
(3) *TEST 11 TEST THAT RCVRDONE IS CLEARED BY READING RBUF
(3) *****
(2) 003566 000004 TST11: SCOPE
(2)
(1) 003570 012737 000010 001160 MOV #10,$TIMES ;;DO 10 ITERATIONS
(2) 003576 012737 000011 001200 MOV #11,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9363
9368 003604 IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
(6) 003604 032737 000020 001220 BIT #WRAP,$USWR
(8) 003612 001404 BEQ $45
(6) 003614 023727 013604 000001 CMP CONSOL,#TRUE
(9) 003622 001001 BNE $46
(6) 003624 $45:
9369 003624 EXIT
(3) 003624 000432 BR TST12 ;;EXIT THIS TEST
9370 003626 ENDIF
(4) 003626 $46:
9371
9372 003626 BGNSUB
(5) 003626 012737 003634 001110 MOV #64,$LPERR
9373 ; OUTPUT A CHARACTER AND WAIT FOR RCVRDONE TO SET.
9374
9375 ; OUTPUT A CHARACTER
9376 003634 LET @TBUF :B= #0
(4) 003634 105077 175426 CLRB @TBUF
9377 ; WAIT MAXIMUM OF 500 MSEC
9378 ; FOR RCVRDONE TO SET IN
9379 ; RCSR
9380 CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
(3) 003640
(7) 003642 010546 MOV R5,-(SP)
(6) 003646 012745 177777 MOV #SET,-(R5)
(5) 003652 013745 001260 MOV RCSR,-(R5)
(4) 003656 012745 000200 MOV #RCVRDONE,-(R5)
(3) 003662 012745 000500 MOV #500,-(R5)
(3) 003666 004737 012446 JSR PC,TIMER
(3) 003666 012605 MOV (SP)+,R5
9381
9382 003670 ; DID IT BECOME READY?
(6) 003670 103001 BCC $47 IF.ERROR THEN
9383 ;RCVRDONE DID NOT SET IN RCSR
9384 003672 ERRHRD 73
(1) 003672 104073 ERROR 73
9385 ; SET IT BACK TO CONTINUE
9386 003674 ENDIF
(4) 003674 $47:
9387 003674 ENDSUB
9388
9389 ; NOW THAT IT IS SET LETS SEE IF READING THE
9390 ; BUFFER CLEARS RCVRDONE.
9391
9392 ;READ BUFFER
9393 003674 LET R0 :B= @RBUF
(4) 003674 117700 175362 MOVB @RBUF,R0
  
```

```
9394  
9395 003700  
  (6) 003700 032777 000200 175352      BIT      #RCVRDONE,@RCSR  
  (9) 003706 001401                      BEQ      $50  
9396  
9397 003710  
  (1) 003710 104074                      ERROR   74  
9398  
9399 003712  
  (4) 003712  
                                $50:
```

IF #RCVRDONE SETIN @RCSR THEN
:RCVRDONE DID NOT CLEAR IN RCSR
ERRHRD 74
: SET IT BACK TO CONTINUE
ENDIF


```

9437 004022 ERRHRD 16
(1) 004022 104016 ERROR 16
9438 004024 ENDIF
(4) 004024 $55:
9439
9440 ;OUTPUT 2ND CHARACTER
9441 004024 LET @TBUF :B= #0
(4) 004024 105077 175236 CLRB @TBUF
9442 ;LET OVERRUN HAPPEN
9443 004030 WAITMS 300.
(4) 004030 010546 MOV R5, -(SP)
(5) 004032 012745 000454 MOV #300, -(R5)
(4) 004036 004737 012620 JSR PC, WAIT
(4) 004042 012605 MOV (SP)+, R5
9444
9445 ;READ BUFFER AND ERROR BITS
9446 004044 LET R4 := @RBUF
(4) 004044 017704 175212 MOV @RBUF, R4
9447
9448 ;IT DIDN'T SET
9449 004050 IF #ORERR NOTSET IN R4 THEN
(6) 004050 032704 040000 BIT #ORERR, R4
(9) 004054 001002 BNE $56
9450 ;ORERR DID NOT SET IN RBUF
9451 004056 ERRHRD 101
(1) 004056 104101 ERROR 101
9452
9453 ;NO USE COMPOUNDING ERRORS
9454 004060 EXIT
(3) 004060 000460 BR TST13 ;:EXIT THIS TEST
9455 004062 ENDIF
(4) 004062 $56:
9456 004062 ENDSUB
9457
9458 ;NOW SEE IF ERROR BIT SET WITH OVERRUN ERROR:
9459 004062 BGNSUB
(5) 004062 012737 004070 001110 MOV #64$, $LPERR
9460 004070 IF #ERROR NOTSET IN R4 THEN
(6) 004070 032704 100000 BIT #ERROR, R4
(9) 004074 001002 BNE $57
9461
9462 ;ERROR DID NOT SET IN RBUF
9463 004076 ERRHRD 102
(1) 004076 104102 ERROR 102
9464
9465 ;-WHEN ORERR SET.
9466 ;GET OUT NOW.
9467 004100 EXIT
(3) 004100 000450 BR TST13 ;:EXIT THIS TEST
9468 004102 ENDIF
(4) 004102 $57:
9469 004102 ENDSUB
9470
9471 004102 BGNSUB
(5) 004102 012737 004110 001110 MOV #64$, $LPERR
9472 ;CHECK REAL RBUF TO SEE IF ORERR IS STILL SET.
  
```

```

9473
9474 004110                                IF #ORERR NOTSETIN @RBUF THEN
(6) 004110 032777 040000 175144          BIT    #ORERR,@RBUF
(9) 004116 001002                                BNE    $60
9475
9476                                ;READING RBUF CLEARED ORERR.
9477 004120                                ERRHRD 103
(1) 004120 104103          ERROR    103
9478                                ;SKIP REST OF TEST
9479 004122                                EXIT
(3) 004122 000437          BR      TST13          ;;EXIT THIS TEST
9480 004124                                ENDIF
(4) 004124                                $60:
9481 004124                                ENDSUB
9482                                BGNSUB
9483 004124                                MOV    #64$, $LPERR
(5) 004124 012737 004132 001110          ;READING RBUF ABOVE SHOULD ENABLE ERROR TO BE CLEARED
9484                                ;BY NEXT TRANSFER.
9485                                ;NOW SEE IF THEY CLEAR WHEN ANOTHER CHAR. IS RECEIVED
9486
9487                                ;SEND A CHARACTER AROUND.
9488                                LET @TBUF :B= #0
9489 004132                                CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
(4) 004132 105077 175130          CLRB  @TBUF
9490 004136                                MOV    R5,-(SP)
(3) 004136 010546          MOV    #SET,-(R5)
(7) 004140 012745 177777          MOV    RCSR,-(R5)
(6) 004144 013745 001260          MOV    #RCVRDONE,-(R5)
(5) 004150 012745 000200          MOV    #500,-(R5)
(4) 004154 012745 000500          JSR   PC,TIMER
(3) 004160 004737 012446          MOV    (SP)+,R5
(3) 004164 012605
9491
9492                                ;DID IT SET IN TIME?
9493 004166                                IF.ERROR THEN
(6) 004166 103001          BCC   $61
9494                                ;RCVRDONE DID NOT SET IN RCSR
9495 004170                                ERRHRD 20
(1) 004170 104020          ERROR    20
9496 004172                                ENDIF
(4) 004172                                $61:
9497
9498                                IF #ORERR SETIN @RBUF THEN
(6) 004172 032777 040000 175062          BIT    #ORERR,@RBUF
(9) 004200 001402                                BEQ    $62
9499                                ;ORERR DID NOT CLEAR IN RBUF
9500 004202                                ERRHRD 104
(1) 004202 104104          ERROR    104
9501
9502                                ;-AFTER RECEIVING ANOTHER CHAR
9503                                ;SKIP AROUND REST
9504 004204                                EXIT
(3) 004204 000406          BR      TST13          ;;EXIT THIS TEST
9505 004206                                ENDIF
(4) 004206                                $62:
9506
  
```



```

CNDLAAO DLV11-J TEST MACY11 30(1046) 23-DEC-82 13:56 PAGE 172-3K 4
CNDLAA.P11 23-DEC-82 13:55 T12 TEST THE OVERRUN & ERROR BITS - RBUF 14 SEQ 0049

9507 004206 IF #ERROR SETIN @RBUF THEN
(6) 004206 032777 100000 175046 BIT #ERROR,@RBUF
(9) 004214 001401 BEQ $63
9508 :ERROR DID NOT CLEAR IN RBUF
9509 004216 ERRHRD 105
(1) 004216 104105 ERROR 105
9510
9511 004220 ENDIF
(4) 004220 $63:
9512 004220 ENDSUB
9513 004220 EXIT
(3) 004220 000400 BR TST13 ;;EXIT THIS TEST

```

```

9515
9516
9528
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 004222 000004
(2)
(1) 004224 012737 000010 001160 MOV #10,$TIMES ;;DO 10 ITERATIONS
(2) 004232 012737 000013 001200 MOV #13,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9533
9534 004240 005037 013616 CLR INTFLAG ;CLEAR 'INTERRUPT OCCURED' FLAG
(4) 004240 005037 013616 LET INTFLAG := #0
9535
9536
9537 004244 013703 001256 MOV DLVEC,R3 ;GET VECTOR ADDRESS
(4) 004244 013703 001256 LET R3 := DLVEC
9538
9539 004250 062703 000004 ADD #4,R3 ;FOR THE TRANSMITTER
(7) 004250 062703 000004 LET R3 := R3 + #4
9540
9541 004254 012723 013610 MOV #INTSRV,(R3)+ ;SET VECTOR TO POINT TO TRANS.SRV AT PRI
9542 004260 012713 000300 MOV #PR6,(R3)
9543 004264 012737 004272 001110 MOV #64$,$LPERR BGNSUB
(5) 004264 012737 004272 001110 ;; MAKE SURE THAT TRANSMITTER READY IS SET
9544
9545 004272 010546 MOV R5,-(SP) CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>
(3) 004272 010546 MOV #SET,-(R5)
(7) 004274 012745 177777 MOV TCSR,-(R5)
(6) 004300 013745 001264 MOV #XMITRDY,-(R5)
(5) 004304 012745 000200 MOV #500,-(R5)
(4) 004310 012745 000500 JSR PC,TIMER
(3) 004314 004737 012446 MOV (SP)+,R5
(3) 004320 012605
9546
9547
9548 004322 042777 000100 174734 BIC #XMITIE,@TCSR ;CLEAR INTERRUPT ENABLE
(7) 004322 042777 000100 174734 LET @TCSR := @TCSR CLR.BY #XMITIE
9549
9550
9551 004330 SETPRI #PRO ;SET IT TO 0
9552
9553
9554 004342 052777 000100 174714 BIS #XMITIE,@TCSR ;NOW SET I.E. BIT
(7) 004342 052777 000100 174714 LET @TCSR := @TCSR SET.BY #XMITIE
9555
9556 004350 CALL TIMER IN <#500,#1,#INTFLAG,#SET>
(3) 004350 010546 MOV R5,-(SP)
(7) 004352 012745 177777 MOV #SET,-(R5)
(6) 004356 012745 013616 MOV #INTFLAG,-(R5)
(5) 004362 012745 000001 MOV #1,-(R5)
(4) 004366 012745 000500 MOV #500,-(R5)
  
```

```

*****
*TEST 13 TRANSMITTER INTERRUPT LOGIC TEST
* LOGICALLY THIS IS 4 SEPARATE TESTS
* A) DOES TRANSMITTER INTERRUPT LOGIC WORK
* B) AT PRIORITY OF 0
* C) AND ONLY ONCE
* D) BUT NOT WITH INTERRUPT ENABLE CLEAR
*****
  
```

TST13: SCOPE

```

(3) 004372 004737 012446      JSR   PC,TIMER
(3) 004376 012605      MOV   (SP)+,R5
9557
9558                                ;DID IT SET IN TIME?
9559 004400                                IF.ERROR THEN
(6) 004400 103001      BCC   $64
9560                                ;INTERRUPT DID NOT OCCUR
9561 004402                                ERRHRD 106
(1) 004402 104106      ERROR  106
9562 004404                                ENDIF
(4) 004404                                $64:
9563
9564                                ;LET POSSIBLE 2'ND INTERR OCCUR
9565 004404                                WAITMS 500.
(4) 004404 010546      MOV   R5, -(SP)
(5) 004406 012745 000764      MOV   #500, -(R5)
(4) 004412 004737 012620      JSR   PC,WAIT
(4) 004416 012605      MOV   (SP)+,R5
9566
9567                                ;DID EXACTLY 1 INTERRUPT OCCUR
9568 004420                                IF INTFLAG GT #1 THEN
(6) 004420 023727 013616 000001      CMP   INTFLAG,#1
(9) 004426 003401      BLE   $65
9569                                ;TRANSMITTER INTERRUPTED TWICE
9570 004430                                ERRHRD 107
(1) 004430 104107      ERROR  107
9571 004432                                ENDIF
(4) 004432                                $65:
9572 004432                                ENDSUB
9573                                ;INTERRUPT WITHOUT INTERRUPT ENABLE SET
9574 004432                                BGNSUB
(5) 004432 012737 004440 001110      MOV   #64$, $LPERR
9575                                ;CLEAR INTERRUPT ENABLE
9576 004440                                LET @TCSR := @TCSR CLR.BY #XMITIE
(7) 004440 042777 000100 174616      BIC   #XMITIE,@TCSR
9577                                ;CLEAR 'INTERRUPT OCCURED' FLAG
9578 004446                                LET INTFLAG := #0
(4) 004446 005037 013616      CLR   INTFLAG
9579                                ;NO INTERRUPTS SHOULD OCCUR, PSW STILL AT 0.
9580                                ;DARE IT TO HAPPEN
9581 004452                                LET @TBUF := #0
(4) 004452 005077 174610      CLR   @TBUF
9582                                ;SEE IF INT FLAG EVER SETS
9583 004456                                CALL TIMER IN <#1000,#1,#INTFLAG,#SET>
(3) 004456 010546      MOV   R5, -(SP)
(7) 004460 012745 177777      MOV   #SET, -(R5)
(6) 004464 012745 013616      MOV   #INTFLAG, -(R5)
(5) 004470 012745 000001      MOV   #1, -(R5)
(4) 004474 012745 001000      MOV   #1000, -(R5)
(3) 004500 004737 012446      JSR   PC,TIMER
(3) 004504 012605      MOV   (SP)+,R5
9584
9585 004506 103401      BCS   1$
9586                                ;DID IT SET DURING TIMER?
9587 004510                                ;BR IF NO
(1) 004510 104110      ERROR  110
                                ;INTERR STILL OCCURED WITH IE DISABLED
                                ERRHRD 110

```


CNDLAAO DLV11-J TEST MACY11 30(1046) 23-DEC-82 13:56 PAGE 173-2^{N 4}
CNDLAA.P11 23-DEC-82 13:55 T13 TRANSMITTER INTERRUPT LOGIC TEST

SEQ 0052

9588 004512

1\$:

```

9590
9591
9601          ;*****
(3)          ;*TEST 14      RECEIVER INTERRUPT LOGIC TEST
(4)          ;*
(4)          ;*
(4)          ;*
(3)          ;*
(2) 004512 000004          TST14: SCOPE
(2)
(1) 004514 012737 000010 001160          MOV #10,$TIMES          ;;DO 10 ITERATIONS
(2) 004522 012737 000014 001200          MOV #14,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
9606 004530          IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
(6) 004530 032737 000020 001220          BIT #WRAP,$USWR
(8) 004536 001404          BEQ $66
(6) 004540 023727 013604 000001          CMP CONSOL,#TRUE
(9) 004546 001002          BNE $67
(6) 004550          $66:
9607 004550 000137 005046          JMP TST15          ; EXIT TEST
9608 004554          $67:          ; ENDIF
(4) 004554
9609
9610          ;CLEAR INTERRUPT OCCURED FLAG
9611          ;SET UP RECEIVER INTER.VECTOR
9612 004554          SETVEC DLVEC,#INTSRV,#PR6
(5) 004554 010146          MOV R1,-(SP)
(5) 004556 013701 001256          MOV DLVEC,R1
(5) 004562 012721 013610          MOV #INTSRV,(R1)+
(5) 004566 012711 000300          MOV #PR6,(R1)
(5) 004572 012601          MOV (SP)+,R1
9613          ;PRIORITY 0 AND MULTIPLE INTERRUPT TEST.-RCVRIE
9614 004574          BGNSUB
(5) 004574 012737 004602 001110          MOV #64,$SLPERR
9615 004602          LET INTFLAG := #0
(4) 004602 005037 013616          CLR INTFLAG
9616          ;CLEAR INTERRUPTS
9617 004606          LET @RCSR := @RCSR CLR.BY #RCVRIE
(7) 004606 042777 000100 174444          BIC #RCVRIE,@RCSR
9618          ;CHANGE PRIORITY
9619          ;...TO 0
9620 004614          SETPRI #PRO
9621
9622          ;SEND A CHARACTER
9623 004626          LET @TBUF :B= #0
(4) 004626 105077 174434          CLRB @TBUF
9624          ;WAIT A MAXIMUM
9625          ;OF 50 MSEC FOR
9626          ;XMIT RDY TO SET IN TCSR
9627 004632          CALL TIMER IN <#500,#RCVRDONE,TCSR,#SET>
(3) 004632 010546          MOV R5,-(SP)
(7) 004634 012745 177777          MOV #SET,-(R5)
(6) 004640 013745 001264          MOV TCSR,-(R5)
(5) 004644 012745 000200          MOV #RCVRDONE,-(R5)
(4) 004650 012745 000500          MOV #500,-(R5)
(3) 004654 004737 012446          JSR PC,TIMER
(3) 004660 012605          MOV (SP)+,R5
    
```



```

9662 004772          LET @TBUF :B= #0
(4)  004772 105077 174270      CLRB  @TBUF
9663                                     ; DARE IT
9664                                     CALL TIMER IN <#500,#1,#INTFLAG,#CLR>
9665 004776          MOV  R5,-(SP)
(3)  004776 010546      MOV  #CLR,-(R5)
(7)  005000 012745 000000      MOV  #INTFLAG,-(R5)
(6)  005004 012745 013616      MOV  #1,-(R5)
(5)  005010 012745 000001      MOV  #500,-(R5)
(4)  005014 012745 000500      JSR  PC,TIMER
(3)  005020 004737 012446      MOV  (SP)+,R5
(3)  005024 012605
9666                                     ;DID IT CLEAR IN TIME?
9667 005026          IF.ERROR THEN
(6)  005026 103001      BCC  $72
9668                                     ;INTERR STILL OCCURED WITH IE DISABLED
9669 005030          ERRHRD 113
(1)  005030 104113      ERROR 113
9670 005032          ENDIF
(4)  005032          $72: .
9671 005032          SETPRI #PR6 ;RAISE CPU PRIORITY
9672 005032          ENDSUB
9673                                     ;CLEAR THE WORLD
9674 005044          RESET      BRFSET
(1)  005044 000005
9675
9676

```

```

9678
9679
9684 *****
(3) *TEST 15 TEST DATA WRAP AROUND: FLAG MODE *****
(3) *****
(2) 005046 000004 TST15: SCOPE
(2)
(1) 005050 012737 000001 001160 MOV #1,$TIMES ;;DO 1 ITERATION
(2) 005056 012737 000015 001200 MOV #15,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9689 005064 032737 000020 001220 BIT #WRAP,$USWR IF #WRAP NOTSET IN $USWR OR CONSOL EQ #TRUE THEN
(6) 005064 001404 BEQ $73
(8) 005072 023727 013604 000001 CMP CONSOL,#TRUE
(6) 005074 001001 BNE $74
(9) 005102
(6) 005104 $73:
9690
9691 005104 BR TST16 EXIT ;;CAN'T TEST WITHOUT A WRAP
(3) 005104 000506 BR TST16 ;;EXIT THIS TEST
9692 005106 $74: ENDF
(4) 005106 BRESET
9693 005106 000005 RESET
9694
9695 005110 INC R2 ;;BINARY COUNT PATTERN
(4) 005110 005002 CLR R2 INCR R2 FROM #0 TO #377 BY #1
(5) 005112 000401 BR $75
(4) 005114 $76:
(8) 005114 005202 INC R2
(5) 005116 $75:
(5) 005116 020227 000377 CMP R2,#377
(7) 005122 003077 BGT $77
9696
9697
9698
9699
9700 005124 CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>
(3) 005124 010546 MOV R5,-(SP)
(7) 005126 012745 177777 MOV #SET,-(R5)
(6) 005132 013745 001264 MOV TCSR,-(R5)
(5) 005136 012745 000200 MOV #XMITRDY,-(R5)
(4) 005142 012745 000500 MOV #500,-(R5)
(3) 005146 004737 012446 JSR PC,TIMER
(3) 005152 012605 MOV (SP)+,R5
9701 005154 IF.ERROR THEN
(6) 005154 103002 BCC $100
9702
9703 005156 ;; TRANSMITTER NEVER BECAME READY
(1) 005156 104123 ERROR 123 ERRHRD 123
9704 005160 EXIT
(3) 005160 000460 BR TST16 ;;EXIT THIS TEST
9705 005162 ENDF
(4) 005162 $100:
9706
9707
9708 005162 ;;START IT ON ITS WAY
(4) 005162 110277 174100 MOVB R2,@TBUF LET @TBUF :B= R2
  
```

```

9709
9710 005166
(3) 005166 010546
(7) 005170 012745 177777
(6) 005174 013745 001260
(5) 005200 012745 000200
(4) 005204 012745 000500
(3) 005210 004737 012446
(3) 005214 012605
9711 005216
(6) 005216 103002
9712 005220
(1) 005220 104124
9713
9714 005222
(3) 005222 000437
9715 005224
(4) 005224 $101:
9716
9717
9718 005224
(4) 005224 017703 174032
9719
9720 005230
(6) 005230 032703 100000
(9) 005234 001401
9721
9722 005236
(1) 005236 104200
9723 005240
(4) 005240 $102:
9724
9725
9726
9727 005240
(6) 005240 020302
(9) 005242 001415
9728 005244
(6) 005244 032737 000001 001220
(9) 005252 001006
9729 005254
(6) 005254 020227 000200
(9) 005260 002002
9730
9731 005262
(1) 005262 104117
9732 005264
(3) 005264 000416
9733 005266
(4) 005266 $105:
9734 005266
(4) 005266 000402
(3) 005270 $104:
9735
9736 005270
(1) 005270 104017
  
```

```

; NOW WAIT FOR RECIEVER DONE
CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
  
```

```

MOV R5, -(SP)
MOV #SET, -(R5)
MOV RCSR, -(R5)
MOV #RCVRDONE, -(R5)
MOV #500, -(R5)
JSR PC, TIMER
MOV (SP)+, R5
  
```

```

IF.ERROR THEN
  ERRHRD 124
; RECIEVER NEVER BECAME READY
EXIT
;:EXIT THIS TEST
ENDIF
  
```

```

;RETRIEVE
LET R3 := @RBUF
;CHECK FOR ERROR DURING TRANSFER
IF #ERROR SETIN R3 THEN
  ;ERROR BIT SET IN HIGH BYTE OF RBUF
  ERRHRD 200
ENDIF
  
```

```

;COMPARE DATA
IF R3 NE R2 THEN
  IF #BIT0 NOTSETIN $USWR THEN
    IF R2 LT #200 THEN
  
```

```

;DATA COMPARE ERR IN 7 BIT WORD
ERRHRD 117
  
```

```

EXIT
;:EXIT THIS TEST
ENDIF
ELSE
  
```

```

;DATA COMP ERR IN 8 BIT WORD
ERRHRD 17
  
```



```

9737 005272
(3) 005272 000413 BR TST16 ::EXIT THIS TEST EXIT
9738 005274 $106: ENDF
(4) 005274
9739 005274 000411 BR $107 ELSE IF #BIT0 NOTSETIN $USWR AND R2 GT #177 THEN
(4) 005274
(3) 005276 $103: BIT #BIT0,$USWR
(7) 005276 032737 000001 001220 BNE $110
(10) 005304 001005 CMP R2,#177
(7) 005306 020227 000177 BLE $110
(10) 005312 003402
9740 ;GETTING 8 BITS ON 7 BIT XMIT
9741 ;MAKE SURE $USWR SETUP CORRECTLY.
9742 005314 104022 ERROR 22 ERRHRD 22
(1) 005314
9743 005316 000401 BR TST16 ::EXIT THIS TEST EXIT
(3) 005316 ENDF
9744 005320 $110:
(4) 005320 $107:
(4) 005320
9745 005320 000675 BR $76 ENDINC ;R2
(4) 005320
(3) 005322 $77:
  
```

```

9747
9748
9753
(3)
(3)
(2) 005322 000004
(2)
(1) 005324 012737 000001 001160 MOV #1,$TIMES ;;DO 1 ITERATION
(2) 005332 012737 000016 001200 MOV #16,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
9758 005340 032737 000020 001220 BIT #WRAP,$USWR IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
(6) 005340 001404 BEQ $111
(8) 005346 023727 013604 000001 CMP CONSOL,#TRUE
(6) 005350 001002 BNE $112
(9) 005360 000137 005772 $111: JMP TST17 ;EXIT TEST
9759 005364 ENDIF
(4) 005364 $112:
9761
9762 ; THIS TEST WILL RUN BOTH TRANSMITTER AND
9763 ; RECIEVER AT FULL SPEED TESTING
9764 ; THE ABILITY OF THE MODULE
9765 ; TO HANDLE INTERRUPTS FROM BOTH SIDES AT ONCE.
9766
9767 ;DOUBLE BUFFERING IS NOT FULLY TESTED BECAUSE OF
9768 ;APT CONSIDERATIONS. I.E. 'BREAK' FROM APT
9769 ;CAUSES OVERRUN ERRORS. THEREFORE TRANSMIT INTR IS
9770 ;ENABLED ONLY AFTER THE RECVR HAS OBTAINED THE LAST WORD
9771
9772 ; THIS TEST WILL TRANSFER A MAXIMUM OF 400(8)
9773 ; CHARACTERS THROUGH THE MODULE, BUT IF AN ERROR
9774 ; IS DETECTED BY THE TEST A PREMATURE SHUTDOWN OCCURS.
9775
9776 ;CHANGE PRIORITY
9777 ;..TO 0
9778 005364 SETPRI #PRO
9779 ;GET VECTOR ADDRESS
9780 005376 LET R1 := DLVEC
(4) 005376 013701 001256 MOV DLVEC,R1
9781 ;RCVR VECTOR
9782 005402 LET (R1)+ := #REC
(4) 005402 012721 005664 MOV #REC,(R1)+
9783 005406 LET (R1)+ := #PR6
(4) 005406 012721 000300 MOV #PR6,(R1)+
9784 ;POINT TO TRANSMITTER VECTOR
9785 ;AND SET IT UP ALSO
9786 005412 LET (R1)+ := #TRAN
(4) 005412 012721 005626 MOV #TRAN,(R1)+
9787 005416 LET (R1) := #PR6
(4) 005416 012711 000300 MOV #PR6,(R1)
9788
9789 ; CLEAR ERROR COUNTER
9790 005422 LET ERRCNT := #0
(4) 005422 005037 005622 CLR ERRCNT
9791 005426 LET DATA := #0 ;XMIT DATA
(4) 005426 005037 005662 CLR DATA
  
```

| | | | | | | | |
|------|--------|--------|--------|--------|-------|---------------|--------------------------------------|
| 9792 | 005432 | | | | | | IF #BIT0 NOTSETIN \$USWR THEN |
| (6) | 005432 | 032737 | 000001 | 001220 | BIT | #BIT0,\$USWR | |
| (9) | 005440 | 001004 | | | BNE | \$113 | |
| 9793 | 005442 | | | | | | LET NUMBER := #200 |
| (4) | 005442 | 012737 | 000200 | 005624 | MOV | #200,NUMBER | |
| 9794 | 005450 | | | | | | ELSE |
| (4) | 005450 | 000403 | | | BR | \$114 | |
| (3) | 005452 | | | | | \$113: | |
| 9795 | 005452 | | | | | | LET NUMBER := #400 |
| (4) | 005452 | 012737 | 000400 | 005624 | MOV | #400,NUMBER | |
| 9796 | 005460 | | | | | | ENDIF |
| (4) | 005460 | | | | | \$114: | |
| 9797 | | | | | | | |
| 9798 | 005460 | | | | | | BRESET ;SET UP ALL REGISTERS |
| (1) | 005460 | 000005 | | | RESET | | |
| 9799 | | | | | | | |
| 9800 | | | | | | | ;SET I.E. IN TRANSMITTER |
| 9801 | 005462 | | | | | | LET @TCSR := @TCSR SET.BY #XMITIE |
| (7) | 005462 | 052777 | 000100 | 173574 | BIS | #XMITIE,@TCSR | |
| 9802 | | | | | | | ;AND RECEIVER |
| 9803 | 005470 | | | | | | LET @RCSR := @RCSR SET.BY #RCVRIE |
| (7) | 005470 | 052777 | 000100 | 173562 | BIS | #RCVRIE,@RCSR | |
| 9804 | | | | | | | |
| 9805 | | | | | | | |
| 9806 | | | | | | | ;NOW WE WAIT |
| 9807 | 005476 | | | | | | REPEAT |
| (3) | 005476 | | | | | \$115: | |
| 9808 | 005476 | | | | | | UNTIL DATA EQ NUMBER OR ERRCNT GT #0 |
| (4) | 005476 | 023737 | 005662 | 005624 | CMP | DATA,NUMBER | |
| (6) | 005504 | 001403 | | | BEQ | \$116 | |
| (4) | 005506 | 005737 | 005622 | | TST | ERRCNT | |
| (7) | 005512 | 003771 | | | BLE | \$115 | |
| (4) | 005514 | | | | | \$116: | |
| 9809 | | | | | | | |
| 9810 | | | | | | | ;NOW LETS CHECK. |
| 9811 | | | | | | | ;TURN OFF ALL INTR ENABLE |
| 9812 | 005514 | | | | | | LET @TCSR := @TCSR CLR.BY #XMITIE |
| (7) | 005514 | 042777 | 000100 | 173542 | BIC | #XMITIE,@TCSR | |
| 9813 | 005522 | | | | | | LET @RCSR := @RCSR CLR.BY #RCVRIE |
| (7) | 005522 | 042777 | 000100 | 173530 | BIC | #RCVRIE,@RCSR | |
| 9814 | 005530 | | | | | | IF ERRCNT NE #0 THEN |
| (6) | 005530 | 005737 | 005622 | | TST | ERRCNT | |
| (9) | 005534 | 001431 | | | BEQ | \$117 | |
| 9815 | 005536 | | | | | | IF #ERROR SETIN RHLD THEN |
| (6) | 005536 | 032737 | 100000 | 005770 | BIT | #ERROR,RHLD | |
| (9) | 005544 | 001424 | | | BEQ | \$120 | |
| 9816 | 005546 | | | | | | IF #ORERR SETIN RHLD THEN |
| (6) | 005546 | 032737 | 040000 | 005770 | BIT | #ORERR,RHLD | |
| (9) | 005554 | 001402 | | | BEQ | \$121 | |
| 9817 | | | | | | | ;OVERRUN ERROR |
| 9818 | 005556 | | | | | | ERRHRD 220 |
| (1) | 005556 | 104220 | | | ERROR | 220 | |
| 9819 | 005560 | | | | | | ELSE IF #FRERR SETIN RHLD THEN |
| (4) | 005560 | 000415 | | | BR | \$122 | |
| (3) | 005562 | | | | | \$121: | |
| (7) | 005562 | 032737 | 020000 | 005770 | BIT | #FRERR,RHLD | |


```

(10) 005570 001402          BEQ      $123
9820                                     ;FRAMING ERROR
9821 005572 104221          ERROR    221          ERRHRD 221
(1) 005572 104221          ERROR    221          ELSE IF #PERR SETIN RHL D THEN
9822 005574 000407          BR       $124
(4) 005574 000407          BR       $124
(3) 005576 032737 010000 005770 $123:  BIT    #PERR,RHL D
(7) 005576 001402          BEQ      $125
(10) 005604 001402          BEQ      $125
9823                                     ;PARITY ERROR
9824 005606 104222          ERROR    222          ERRHRD 222
(1) 005606 104222          ERROR    222          ELSE
9825 005610 000401          BR       $126
(4) 005610 000401          BR       $126
(3) 005612 000401          BR       $125
9826                                     ;UNKNOWN ERROR
9827 005612 104024          ERROR    24          ERRHRD 24
(1) 005612 104024          ERROR    24          ENDIF
9828 005614 000401          BR       $126
(4) 005614 000401          BR       $124
(4) 005614 000401          BR       $122
9829 005614 000401          BR       $127
(4) 005614 000401          BR       $127
(3) 005616 000401          BR       $120
9830                                     ;DATA COMPARE ERROR
9831 005616 104120          ERROR    120          ERRHRD 120
(1) 005616 104120          ERROR    120          ENDIF
9832 005620 000401          BR       $127
(4) 005620 000401          BR       $127
9833 005620 000401          BR       $117
(4) 005620 000401          BR       $117
9834 005620 000464          BR       TST17          EXIT ;SKIP OVER SUPPORT ROUTINES & STORAGE
(3) 005620 000464          BR       TST17          ;;EXIT THIS TEST
9835
9836
9837 005622 000000          ERRCNT: 0
9838 005624 000000          NUMBER: 0
    
```

```

9840
9841
9846          ;:*****
9847          ;:TRANSMIT INTERRUPT HANDLER
9848          ;:*****
9853 005626          BGNSRV TRAN
(1) 005626          TRAN:
9854 005626          IF DATA NE NUMBER AND ERRCNT EQ #0 THEN
(6) 005626 023737 005662 005624          CMP DATA,NUMBER
(9) 005634 001406          BEQ $130
(6) 005636 005737 005622          TST  ERRCNT
(9) 005642 001003          BNE  $130
9855          ;SHIP OUT WORD
9856 005644          LET @TBUF := DATA
(4) 005644 013777 005662 173414          MOV  DATA,@TBUF
9857 005652          ENDIF
(4) 005652          $130:
9858          ;STOP INTERR, NOT EXER DOUBL BUFFER
9859 005652          LET @TCSR := @TCSR CLR.BY #XMITIE
(7) 005652 042777 000100 173404          BIC  #XMITIE,@TCSR
9860 005660          ENDSRV
(1) 005660 000002          RTI
9861
9862 005662 000000          DATA: 0
9863
9868          ;:*****
9869          ;:RECEIVER INTERRUPT HANDLER
9870          ;:*****
9875 005664          BGNSRV REC
(1) 005664          REC:
9876          ;GET CHAR
9877 005664          LET RHL := @RBUF
(4) 005664 017737 173372 005770          MOV  @RBUF,RHL
9878          ;CHECK ERROR
9879 005672          IF #ERROR SETIN RHL OR RHL NE DATA THEN
(6) 005672 032737 100000 005770          BIT  #ERROR,RHL
(8) 005700 001004          BNE  $131
(6) 005702 023737 005770 005662          CMP  RHL,DATA
(9) 005710 001411          BEQ  $132
(6) 005712          $131:
9880          ;STOP ALL INTERR PROC & GET OUT
9881 005712          LET DATA := NUMBER
(4) 005712 013737 005624 005662          MOV  NUMBER,DATA
9882 005720          LET @RCSR := @RCSR CLR.BY #RCVRIE
(7) 005720 042777 000100 173332          BIC  #RCVRIE,@RCSR
9883 005726          LET ERRCNT := ERRCNT + #1
(7) 005726 005237 005622          INC  ERRCNT
9884 005732          ELSE
(4) 005732 000415          BR   $133
(3) 005734          $132:
9885 005734          LET DATA := DATA + #1
(7) 005734 005237 005662          INC  DATA
9886 005740          IF DATA EQ NUMBER THEN
(6) 005740 023737 005662 005624          CMP  DATA,NUMBER
(9) 005746 001004          BNE  $134
9887 005750          LET @RCSR := @RCSR CLR.BY #RCVRIE
  
```

```
(7) 005750 042777 000100 173302      BIC  #RCVRIE,@RCSR
9888 005756                                     ELSE
(4) 005756 000403                          BR   $135
(3) 005760      $134:
9889                                     '
9890 005760                                     ;ALLOW NEXT XMIT INTERR
(7) 005760 052777 000100 173276      BIS  #XMITIE,@TCSR      LET @TCSR := @TCSR SET.BY #XMITIE
9891 005766                                     ENDIF
(4) 005766      $135:
9892 005766                                     ENDIF
(4) 005766      $133:
9893 005766                                     ENDSRV
(1) 005766 000002      RTI
9894
9895 005770 000000      RHLD: 0
9896
```


9898
 9899
 9910
 (3)
 (4)
 (4)
 (4)
 (4)
 (3)
 (2) 005772 000004
 (2)
 (1) 005774 012737 000010 001160
 (2) 006002 012737 000017 001200
 9915
 9916
 9917
 9918 006010
 (6) 006010 032737 000020 001220
 (8) 006016 001404
 (6) 006020 032737 000010 001220
 (9) 006026 001003
 (6) 006030
 9919 006030 000137 006502
 9920 006034
 (4) 006034 000406
 (3) 006036
 9921 006036
 (6) 006036 023727 013604 000001
 (9) 006044 001002
 9922
 9923 006046 000137 006502
 9924 006052
 (4) 006052
 9925 006052
 (4) 006052
 9926 006052
 (5) 006052 012737 006060 001110
 9927
 9928 006060
 (4) 006060 005037 006500
 9929
 9930 006064
 (7) 006064 052777 000001 173172
 9931
 9932 006072
 (4) 006072 012777 000125 173166
 9933
 9934 006100
 (3) 006100 010546
 (7) 006102 012745 177777
 (6) 006106 013745 001260
 (5) 006112 012745 000200
 (4) 006116 012745 000500
 (3) 006122 004737 012446
 (3) 006126 012605
 9935 006130

```

*****
*TEST 17      TEST BREAK DETECTION LOGIC
*              TRANSMIT KNOWN CHAR WITH BREAK SET
*              AND COMPARE RECEIVED WITH 0.
*              FRAMING ERROR WILL ALSO BE CHECKED
*              IF ERROR BITS ARE ENABLED.
*****
  
```

```

TST17:  SCOPE
        MOV      #10,$TIMES      ;;DO 10 ITERATIONS
        MOV      #17,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
                                   ;;DONT DO THIS TEST IF 'BREAK' GENERATION
                                   ;;ENABLED, ELSE WILL HALT TO CONSOLE ODT.
                                   ;;DO IF BREAK 'DETECTION' IS ENABLED.
                                   ;;IF #WRAP NOTSETIN $USWR OR #BRK NOTSETIN $USWR THEN
        BIT      #WRAP,$USWR
        BEQ     $136
        BIT     #BRK,$USWR
        BNE     $137
$136:   JMP      TST20          ELSE          ;EXIT TEST
        BR      $140
$137:
        CMP     CONSOL,#TRUE    IF CONSOL EQ #TRUE THEN
        BNE     $141
                                   ;CAN'T TEST CONSOLE
                                   ;EXIT TEST
        JMP     TST20          ENDIF
$141:
$140:
        BGNSUB
        LET     ERRCHK := #0    ; CLEAR ERROR WORD
                                   ;SET BREAK BIT
        LET     @TCSR := @TCSR SET.BY #BREAK
                                   ;NON-ZERO CHAR. '*'
        LET     @TBUF := #125
                                   ;WAIT FOR DONE
        CALL    TIMER IN <#500,#RCVRDONE,RCSR,#SET>
        MOV     R5,-(SP)
        MOV     #SET,-(R5)
        MOV     RCSR,-(R5)
        MOV     #RCVRDONE,-(R5)
        MOV     #500,-(R5)
        JSR    PC,TIMER
        MOV     (SP)+,R5
        IF.ERROR THEN
  
```

```

(6) 006130 103001          BCC      $142
9936
9937 006132          ERROR    115
(1) 006132 104115
9938 006134          $142:
(4) 006134
9939 006134          MOV      @RBUF,R0
(4) 006134 017700 173122
9940 006140          TSTB    R0
(6) 006140 105700          BEQ      $143
(9) 006142 001403
9941
9942 006144          BIS      #BIT0,ERRCHK
(7) 006144 052737 000001 006500
9943 006152          $143:
(4) 006152
9944 006152          BIT      #FRERR,R0
(6) 006152 032700 020000          BNE      $144
(9) 006156 001003
9945 006160          BIS      #BIT1,ERRCHK
(7) 006160 052737 000002 006500
9946 006166          $144:
(4) 006166
9947 006166          BIT      #PARITY,$USWR
(6) 006166 032737 000002 001220          BEQ      $145
(9) 006174 001421
9948
9949 006176          BIT      #EVENODD,$USWR
(6) 006176 032737 000004 001220          BNE      $146
(9) 006204 001007
9950
9951 006206          BIT      #PERR,R0
(6) 006206 032700 010000          BNE      $147
(9) 006212 001003
9952
9953 006214          BIS      #BIT2,ERRCHK
(7) 006214 052737 000004 006500
9954 006222          $147:
(4) 006222
9955 006222          BR      $150
(4) 006222 000406          $146:
(3) 006224
9956 006224          BIT      #PERR,R0
(6) 006224 032700 010000          BEQ      $151
(9) 006230 001403
9957 006232          BIS      #BIT3,ERRCNT
(7) 006232 052737 000010 005622
9958 006240          $151:
(4) 006240
9959 006240          $150:
(4) 006240
9960 006240          $145:
(4) 006240
9961
9962 006240          RESET
(1) 006240 000005
  
```

```

: RECIEVER DONE DID NOT SET
ERRHRD 115
ENDIF
LET R0 := @RBUF
IFB R0 NE #0 THEN
: BREAK DID NOT EQUAL 0
LET ERRCHK := ERRCHK SET.BY #BIT0
ENDIF
IF #FRERR NOTSETIN R0 THEN
LET ERRCHK := ERRCHK SET.BY #BIT1
ENDIF
IF #PARITY SETIN $USWR THEN
: ODD PARITY ENABLED
IF #EVENODD NOTSETIN $USWR THEN
: BREAK SHOULD GENERATE A PARITY ERRO
IF #PERR NOTSETIN R0 THEN
: NO PAR ERROR WHEN THERE SHOULD
LET ERRCHK := ERRCHK SET.BY #BIT2
ENDIF
ELSE
IF #PERR SETIN R0 THEN
LET ERRCNT := ERRCNT SET.BY #BIT3
ENDIF
ENDIF
ENDIF
BRESET ;CLEAN UP
  
```

| | | | | | | | |
|------|--------|--------|--------|--------|--------|---------------|---|
| 9963 | 006242 | 032777 | 170000 | 173012 | BIT | #170000,@RBUF | |
| 9964 | 006250 | 001401 | | | BEQ | 1\$ | |
| 9965 | 006252 | 104033 | | | ERROR | 33 | ;RESET DID NOT CLEAR ERROR,FR ERR,OR PERR IN RBUF |
| 9966 | 006254 | | | | 1\$: | | |
| 9967 | | | | | | | |
| 9968 | 006254 | | | | | | IF #BIT0 SETIN ERRCHK THEN |
| (6) | 006254 | 032737 | 000001 | 006500 | BIT | #BIT0,ERRCHK | |
| (9) | 006262 | 001401 | | | BEQ | \$152 | ERRHRD 121 ;BREAK ERROR |
| 9969 | 006264 | | | | ERROR | 121 | ENDIF |
| (1) | 006264 | 104121 | | | | | IF #BIT1 SETIN ERRCHK THEN |
| 9970 | 006266 | | | | \$152: | | |
| (4) | 006266 | | | | | | |
| 9971 | 006266 | | | | | | ERRHRD 122 ; FRAMING ERROR |
| (6) | 006266 | 032737 | 000002 | 006500 | BIT | #BIT1,ERRCHK | ENDIF |
| (9) | 006274 | 001401 | | | BEQ | \$153 | |
| 9972 | 006276 | | | | ERROR | 122 | |
| (1) | 006276 | 104122 | | | | | |
| 9973 | 006300 | | | | \$153: | | |
| (4) | 006300 | | | | | | IF #BIT2 SETIN ERRCHK THEN |
| 9974 | 006300 | | | | | | ERRHRD 235 |
| (6) | 006300 | 032737 | 000004 | 006500 | BIT | #BIT2,ERRCHK | |
| (9) | 006306 | 001401 | | | BEQ | \$154 | ;NO PARITY ERROR WHEN |
| 9975 | 006310 | | | | ERROR | 235 | ;THERE SHOULD BE |
| (1) | 006310 | 104235 | | | | | ENDIF |
| 9976 | | | | | | | IF #BIT3 SETIN ERRCHK THEN |
| 9977 | | | | | | | ERRHRD 236 |
| 9978 | 006312 | | | | \$154: | | ;PARITY ERROR SHOULD NOT HAVE |
| (4) | 006312 | | | | | | ;OCCURED WITH EVEN PARITY |
| 9979 | 006312 | | | | | | ;ENABLED AND BREAK SET |
| (6) | 006312 | 032737 | 000010 | 006500 | BIT | #BIT3,ERRCHK | ENDIF |
| (9) | 006320 | 001401 | | | BEQ | \$155 | |
| 9980 | 006322 | | | | ERROR | 236 | ENDSUB |
| (1) | 006322 | 104236 | | | | | BGNSUB |
| 9981 | | | | | | | |
| 9982 | | | | | | | ;SET BREAK BIT |
| 9983 | | | | | | | LET @TCSR := @TCSR SET.BY #BREAK |
| 9984 | 006324 | | | | \$155: | | CALL TIMER IN <#500,#BREAK,TCSR,#SET> |
| (4) | 006324 | | | | | | |
| 9985 | 006324 | | | | | | |
| 9986 | 006324 | | | | | | |
| (5) | 006324 | 012737 | 006332 | 001110 | MOV | #64\$,\$LPERR | |
| 9987 | | | | | | | |
| 9988 | 006332 | | | | | | |
| (7) | 006332 | 052777 | 000001 | 172724 | BIS | #BREAK,@TCSR | |
| 9989 | | | | | | | |
| 9990 | 006340 | | | | | | |
| (3) | 006340 | 010546 | | | MOV | R5,-(SP) | |
| (7) | 006342 | 012745 | 177777 | | MOV | #SET,-(R5) | |
| (6) | 006346 | 013745 | 001264 | | MOV | TCSR,-(R5) | |
| (5) | 006352 | 012745 | 000001 | | MOV | #BREAK,-(R5) | |
| (4) | 006356 | 012745 | 000500 | | MOV | #500,-(R5) | |
| (3) | 006362 | 004737 | 012446 | | JSR | PC,TIMER | |
| (3) | 006366 | 012605 | | | MOV | (SP)+,R5 | |
| 9991 | | | | | | | |
| 9992 | | | | | | | ;DID IT SET? |
| 9993 | 006370 | | | | | | IF.ERROR THEN |


```

(6) 006370 103001          BCC      $156
9994
9995 006372                ERROR    21
(1) 006372 104021
9996 006374                ENDIF
(4) 006374          $156:
9997
9998
9999 006374                ;CLEAR BREAK BIT
(7) 006374 042777 000001 172662    BIC      #BREAK,@TCSR
10000 006402                LET @TCSR := @TCSR CLR.BY #BREAK
(4) 006402 010546                WAITMS 100.
(5) 006404 012745 000144          MOV      R5,-(SP)
(4) 006410 004737 012620          MOV      #100,-(R5)
(4) 006414 012605                JSR      PC,WAIT
10001
10002
10003
10004 006416                ;READ RBUF TO CLEAR ERRORS & REC DONE
(4) 006416 017700 172640          MOV      @RBUF,R0
10005
10006 006422                ;SEND CHAR
(4) 006422 012777 000125 172636    MOV      #125,@TBUF
10007
10008 006430                ;WAIT FOR DONE BIT
(3) 006430 010546                CALL    TIMER IN <#500,#RCVRDONE,RCSR,#SET>
(7) 006432 012745 177777          MOV      R5,-(SP)
(6) 006436 013745 001260          MOV      #SET,-(R5)
(5) 006442 012745 000200          MOV      RCSR,-(R5)
(4) 006446 012745 000500          MOV      #RCVRDONE,-(R5)
(3) 006452 004737 012446          MOV      #500,-(R5)
(3) 006456 012605                JSR      PC,TIMER
10009 006460                MOV      (SP)+,R5
(6) 006460 103001          BCC      $157
10010
10011 006462                IF.ERROR THEN
(1) 006462 104230          ERROR    230
10012 006464                ;RECEIVER NEVER CAME READY
(4) 006464          $157:                ERRHRD 230
10013
10014
10015 006464                ENDIF
(6) 006464 127727 172572 000125    CMPB    @RBUF,#125
(9) 006472 001401                IFB @RBUF NE #125 THEN
10016
10017 006474                ;WAS CHAR AFTER BREAK RECEIVED
(1) 006474 104231          ERROR    231
10018 006476                ;CHAR AFTER BREAK NOT RECEIVED CORRECTLY
(4) 006476          $160:                ERRHRD 231
10019 006476                ENDIF
10020 006476                ENDSUB
(3) 006476 000401                EXIT
10021 006500 000000          ERRCHK: BR      TST20
                                ;:EXIT THIS TEST
                                .WORD 0
    
```

```

10023
10024
10029          ;:*****
(3)          ;:TEST 20      NOT A TEST - SEND BACK TO LOOP
(3)          ;:*****
(2) 006502 000004 TST20: SCOPE
(2)
(1) 006504 012737 000001 001160      MOV      #1,$TIMES      ;;DO 1 ITERATION
10034 006512 032777 010000 172420      BIT      #BIT12,@SWR      IF #BIT12 SETIN @SWR THEN
(6) 006512 032777 010000 172420      BEQ      $161
(9) 006520 001513
10035 006522                                IF PHASE2 EQ #TRUE THEN
(6) 006522 023727 014166 000001      CMP      PHASE2,#TRUE
(9) 006530 001021      BNE      $162
10036 006532      TYPTXT <<CRLF>/      ** PHASE 2 SUMMARY **/<CRLF>>
10037 006572                                ELSE
(4) 006572 000425      BR      $163
(3) 006574      $162:                                IF P1CNT EQ #0 THEN
10038 006574                                TST      P1CNT
(6) 006574 005737 014170      BNE      $164
(9) 006600 001022      TYPTXT <<CRLF>/      ** PHASE 1 SUMMARY **/<CRLF>>
10039 006602                                LET P1CNT := P1CNT + #1
10040 006642 005237 014170      INC      P1CNT
(7) 006642
10041 006646      $164:                                ENDIF
(4) 006646
10042 006646      $163:                                ENDIF
(4) 006646
10043 006646      TYPTXT <*,CSR: *>
10044 006662      TYPOCT DLADD
10045 006670      TYPTXT <*,VECTOR: *>
10046 006710      TYPOCT DLVEC
10047 006716      TYPTXT <*,ERRORS: *>
10048 006736      TYPDEC $ERTTL
10049 006744 104401 001171      TYPE      ,$CRLF
10050 006750                                ENDIF
(4) 006750      $161:                                ; RESET FOR NEXT DEVICE/PASS
10051 006750 005037 001112      CLR      $ERTTL      IF PHASE2 EQ #TRUE THEN
10052 006754                                CMP      PHASE2,#TRUE
(6) 006754 023727 014166 000001      BNE      $165
(9) 006762 001004
10053 006764                                LET PHASE2 := #0
(4) 006764 005037 014166      CLR      PHASE2
10054 006770                                LET $UNIT := $UNIT + #1
(7) 006770 005237 001206      INC      $UNIT
10055 006774                                ENDIF
(4) 006774      $165:
10056 006774 000137 001732      JMP      LOOP      ; BACK UP TO THE BEGINNING
  
```

```

10058
10059
10060 007000
10070
(3)
(4)
(4)
(4)
(3)
(2) 007000 000004
(2)
(1) 007002 012737 000001 001160 MOV #1,$TIMES ;;DO 1 ITERATION
(2) 007010 012737 000021 001200 MOV #21,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
10071
10076
10077
10078 007016
(4) 007016 012700 007744 MOV #INTRTABLE,R0 ;CLEAR OUT INTERRUPT TABLE
10079 007022 REPEAT
(3) 007022 $166: LET (R0)+ := #0
10080 007022 (4) 007022 005020 CLR (R0)+ UNTIL R0 EQ #TABEND
10081 007024 (3) 007024 020027 007764 CMP R0,#TABEND
(6) 007030 001374 BNE $166
10082
10083
10084 007032 SETPRI #PR6 ;SET PRIORITY TO 6
10085
10086
10087 007044
(4) 007044 013700 001256 MOV DLVEC,R0 ;SET UP ALL INTERRUPT VECTORS
10088 007050 (4) 007050 012701 007424 MOV #RCVROSRV,R1 LET R1 := #RCVROSRV
10089 007054 (3) 007054 $167: REPEAT
10090 007054 (4) 007054 010120 MOV R1,(R0)+ LET (R0)+ := R1
10091 007056 (4) 007056 012720 000300 MOV #PR6,(R0)+ LET (R0)+ := #PR6
10092 007062 (7) 007062 062701 000030 ADD #30,R1 LET R1 := R1 + #30
10093 007066 (3) 007066 020127 007724 CMP R1,#SRVEND UNTIL R1 EQ #SRVEND
(6) 007072 001370 BNE $167
10094
10095 ;ENABLE INTERRUPTS ON ALL ACTIVE LINES,
10096 ;ALSO, XMIT CHAR'S TO PRIME RECEIVERS
10097
10098 ;COPY MASK
10099 007074 (4) 007074 013737 014162 007764 MOV MASK,CHMASK LET CHMASK := MASK
10100 007102 (4) 007102 005037 007766 CLR CHCNT LET CHCNT := #0
10101 007106 (4) 007106 013700 001254 MOV DLADD,R0 LET R0 := DLADD

```


| | | | | | | | | | |
|-------|--------|--------|--------|--------|--------|-------|-----------------|--|--|
| 10102 | 007112 | | | | | | | | REPEAT |
| (3) | 007112 | | | | | | | | |
| 10103 | 007112 | | | | \$170: | | | | IF #BIT0 SET IN CHMASK THEN |
| (6) | 007112 | 032737 | 000001 | 007764 | | BIT | #BIT0,CHMASK | | |
| (9) | 007120 | 001430 | | | | BEQ | \$171 | | |
| 10104 | | | | | | | | | ;SET RCSR IE |
| 10105 | 007122 | | | | | | | | LET (R0) := (R0) SET.BY #BIT6 |
| (7) | 007122 | 052710 | 000100 | | | BIS | #BIT6,(R0) | | |
| 10106 | 007126 | | | | | | | | LET R0 := R0 + #4 |
| (7) | 007126 | 062700 | 000004 | | | ADD | #4,R0 | | |
| 10107 | | | | | | | | | ;SET XCSR IE |
| 10108 | 007132 | | | | | | | | LET (R0) := (R0) SET.BY #BIT6 |
| (7) | 007132 | 052710 | 000100 | | | BIS | #BIT6,(R0) | | |
| 10109 | | | | | | | | | ;LOAD XBUF |
| 10110 | 007136 | | | | | | | | LET 2(R0) := #252 |
| (4) | 007136 | 012760 | 000252 | 000002 | | MOV | #252,2(R0) | | |
| 10111 | | | | | | | | | ;GO BACK TO RCSR |
| 10112 | 007144 | | | | | | | | LET R0 := R0 - #4 |
| (7) | 007144 | 162700 | 000004 | | | SUB | #4,R0 | | |
| 10113 | | | | | | | | | ;LOOK FOR RCVR DONE |
| 10114 | 007150 | | | | | | | | CALL TIMER IN <#500,#RCVRDONE,R0,#SET> |
| (3) | 007150 | 010546 | | | | MOV | R5,-(SP) | | |
| (7) | 007152 | 012745 | 177777 | | | MOV | #SET,-(R5) | | |
| (6) | 007156 | 010045 | | | | MOV | R0,-(R5) | | |
| (5) | 007160 | 012745 | 000200 | | | MOV | #RCVRDONE,-(R5) | | |
| (4) | 007164 | 012745 | 000500 | | | MOV | #500,-(R5) | | |
| (3) | 007170 | 004737 | 012446 | | | JSR | PC,TIMER | | |
| (3) | 007174 | 012605 | | | | MOV | (SP)+,R5 | | |
| 10115 | | | | | | | | | ;DID IT SET IN TIME? |
| 10116 | 007176 | | | | | | | | IF.ERROR THEN |
| (6) | 007176 | 103001 | | | | BCC | \$172 | | |
| 10117 | | | | | | | | | ;RCVRDONE DID NOT SET IN RCSR |
| 10118 | 007200 | | | | | | | | ERRHRD 23 |
| (1) | 007200 | 104023 | | | | ERROR | 23 | | |
| 10119 | 007202 | | | | | | | | ENDIF |
| (4) | 007202 | | | | \$172: | | | | |
| 10120 | 007202 | | | | \$171: | | | | ENDIF |
| (4) | 007202 | | | | | | | | |
| 10121 | | | | | | | | | |
| 10122 | 007202 | | | | | | | | LET R0 := R0 + #10 |
| (7) | 007202 | 062700 | 000010 | | | ADD | #10,R0 | | |
| 10123 | | | | | | | | | ;SETUP FOR NEXT CHANNEL |
| 10124 | 007206 | | | | | | | | LET CHMASK := CHMASK SHIFT -1 |
| (8) | 007206 | 006237 | 007764 | | | ASR | CHMASK | | |
| 10125 | 007212 | | | | | | | | LET CHCNT := CHCNT + #1 |
| (7) | 007212 | 005237 | 007766 | | | INC | CHCNT | | |
| 10126 | 007216 | | | | | | | | IF CHCNT EQ #3 THEN |
| (6) | 007216 | 023727 | 007766 | 000003 | | CMP | CHCNT,#3 | | |
| (9) | 007224 | 001007 | | | | BNE | \$173 | | |
| 10127 | 007226 | | | | | | | | CALL TSTCON |
| (3) | 007226 | 004737 | 013530 | | | JSR | PC,TSTCON | | |
| 10128 | 007232 | | | | | | | | IF CONSOLE NE #0 THEN |
| (6) | 007232 | 005737 | 013604 | | | TST | CONSOLE | | |
| (9) | 007236 | 001402 | | | | BEQ | \$174 | | |
| 10129 | 007240 | | | | | | | | LET CHCNT := CHCNT + #1 |
| (7) | 007240 | 005237 | 007766 | | | INC | CHCNT | | |

```

10130 007244                                ENDIF
      (4) 007244                                $174:
10131 007244                                ENDIF
      (4) 007244                                $173:
10132 007244                                UNTIL CHCNT EQ #4
      (3) 007244 023727 007766 000004          CMP      CHCNT,#4
      (6) 007252 001317                        BNE      $170
10133
10134
10135
10136
10137
10138
10139
10140
10141
10142
10143
10144 007254                                ;SET UP POINTER FOR SERVICE ROUTINE
      (4) 007254 012704 007744                MOV      #INTRTABLE,R4          LET R4 := #INTRTABLE
10145
10146
10147 007260                                ;LET EM GO
      SETPRI #PRO
10148
10149
10150 007272                                ;DISABLE ALL INTERRUPTS
      (1) 007272 000005                RESET          BRESET
10151
10152 007274                                LET TCNT := #0
      (4) 007274 005037 007772                CLR      TCNT
10153
10154 007300                                ;CLEAR ERR FLAG
      (4) 007300 005037 007770                CLR      PRIERR          LET PRIERR := #0
10155 007304                                LET R0 := #INTRTABLE
      (4) 007304 012700 007744                MOV      #INTRTABLE,R0
10156
10157 007310                                ;SETUP EXPECTED VALUE
      (4) 007310 012701 007746                MOV      #INTRTABLE+2,R1    LET R1 := #INTRTABLE+2
10158 007314 013737 014162 007764          MOV      MASK,CHMASK        ;GET # OF CHANNELS IN TCNT
10159
10160 007322 006237 007764                1$: ASR      CHMASK
10161 007326 103002                        BCC      2$
10162 007330 005237 007772                INC      TCNT                ;GET TOTAL CHANNEL CT
10163 007334 005737 007764                2$: TST      CHMASK
10164 007340 001370                        BNE      1$
10165
10166 007342 004737 013530                JSR      PC,TSTCON          ;CORRECT CHANNEL CT FOR CONSOLE
10167 007346 005737 013604                TST      CONSOLE
10168 007352 001402                        BEQ      3$
10169 007354 005337 007772                DEC      TCNT
10170 007360 006337 007772                3$: ASL      TCNT                ;APPLY X2 MULT FACTOR
10171
10172 007364 005337 007772                4$: DEC      TCNT
10173 007370 001410                        BEQ      5$
10174 007372 023727 007772 177777          CMP      TCNT,#-1          ;ALL 1'S ?
10175 007400 001404                        BEQ      5$                ;BR IF YES
    
```

```
10176 007402 022120          CMP      (R1)+,(R0)+      :R1-R0
10177 007404 003367          BGT      4$              :BR IF R1>R0
10178 007406 005237 007770   INC      PRIERR         :ELSE ERROR
10179
10180 007412          5$:
10181 007412          IF PRIERR NE #0 THEN
   (6) 007412 005737 007770   TST      PRIERR
   (9) 007416 001401          BEQ      $175
10182 007420          ERROR 250
   (1) 007420 104250          ERRHRD 250
10183
10184
10185
10186
10187
10188 007422          :CHANNELS DID NOT INTR ACCORDING TO
   (4) 007422          :ASSIGNED PRIORITY. THE BYTE ENTRIES
10189
10190 007422 000565          :IN INTRTABLE: SHOULD BE IN THE ORDER
   (3) 007422          :THAT THEY APPEAR IN THE CHANNEL #
          :TABLE .(EXCLUDING CHANNELS NOT ACTIVE)
          ENDIF
          BR      TST22
          EXIT
          ;;EXIT THIS TEST
```



```

10192
10193
10198
10199
10200
10205
10206 007424
      (1) 007424
10207
10208 007424
      (4) 007424 013724 007724
10209
10210 007430
      (4) 007430 013737 001260 007774
10211 007436
      (7) 007436 062737 000000 007774
10212
10213 007444
      (7) 007444 042777 000100 000322
10214 007452
      (1) 007452 000002
10215
10216
10217 007454
      (1) 007454
10218
10219 007454
      (4) 007454 013724 007734
10220
10221 007460
      (4) 007460 013737 001260 007774
10222 007466
      (7) 007466 062737 000004 007774
10223
10224 007474
      (7) 007474 042777 000100 000272
10225 007502
      (1) 007502 000002
10226
10227 007504
      (1) 007504
10228
10229 007504
      (4) 007504 013724 007726
10230
10231 007510
      (4) 007510 013737 001260 007774
10232 007516
      (7) 007516 062737 000010 007774
10233
10234 007524
      (7) 007524 042777 000100 000242
10235 007532
      (1) 007532 000002
10236
10237 007534
    
```

```

*****
* START OF SERVICE ROUTINES
*****
    
```

```

RCVROSRV:
BGNSRV RCVROSRV
      :PUT INTR IDENTIFIER IN INTRTABLE
      LET (R4)+ := RCHO
      :GENERATE CSR ADDRESS
      LET TEMP := RCSR
      LET TEMP := TEMP + #0
      :ONE INTR IS ALL WE WANT FROM HERE
      LET @TEMP := @TEMP CLR.BY #BIT6
ENDSRV
    
```

```

XMITOSRV:
BGNSRV XMITOSRV
      :PUT INTR IDENTIFIER IN INTRTABLE
      LET (R4)+ := TCHO
      :GENERATE CSR ADDRESS
      LET TEMP := RCSR
      LET TEMP := TEMP + #4
      :ONE INTR IS ALL WE WANT FROM HERE
      LET @TEMP := @TEMP CLR.BY #BIT6
ENDSRV
    
```

```

RCVR1SRV:
BGNSRV RCVR1SRV
      :PUT INTR IDENTIFIER IN INTRTABLE
      LET (R4)+ := RCH1
      :GENERATE CSR ADDRESS
      LET TEMP := RCSR
      LET TEMP := TEMP + #10
      :ONE INTR IS ALL WE WANT FROM HERE
      LET @TEMP := @TEMP CLR.BY #BIT6
ENDSRV
    
```

```

BGNSRV XMIT1SRV
    
```

J 6

```
(1) 007534 XMIT1SRV:
10238
10239 007534 :PUT INTR IDENTIFIER IN INTRTABLE
(4) 007534 013724 007736 MOV TCH1,(R4)+ LET (R4)+ := TCH1
10240 :GENERATE CSR ADDRESS
10241 007540 LET TEMP := RCSR
(4) 007540 013737 001260 007774 MOV RCSR,TEMP
10242 007546 LET TEMP := TEMP + #14
(7) 007546 062737 000014 007774 ADD #14,TEMP
10243 :ONE INTR IS ALL WE WANT FROM HERE
10244 007554 LET @TEMP := @TEMP CLR.BY #BIT6
(7) 007554 042777 000100 000212 BIC #BIT6,@TEMP
10245 007562 ENDSRV
(1) 007562 000002 RTI
10246
10247 007564 BGNSRV RCVR2SRV
(1) 007564 RCVR2SRV:
10248 :PUT INTR IDENTIFIER IN INTRTABLE
10249 007564 LET (R4)+ := RCH2
(4) 007564 013724 007730 MOV RCH2,(R4)+
10250 :GENERATE CSR ADDRESS
10251 007570 LET TEMP := RCSR
(4) 007570 013737 001260 007774 MOV RCSR,TEMP
10252 007576 LET TEMP := TEMP + #20
(7) 007576 062737 000020 007774 ADD #20,TEMP
10253 :ONE INTR IS ALL WE WANT FROM HERE
10254 007604 LET @TEMP := @TEMP CLR.BY #BIT6
(7) 007604 042777 000100 000162 BIC #BIT6,@TEMP
10255 007612 ENDSRV
(1) 007612 000002 RTI
10256
10257 007614 BGNSRV XMIT2SRV
(1) 007614 XMIT2SRV:
10258 :PUT INTR IDENTIFIER IN INTRTABLE
10259 007614 LET (R4)+ := TCH2
(4) 007614 013724 007740 MOV TCH2,(R4)+
10260 :GENERATE CSR ADDRESS
10261 007620 LET TEMP := RCSR
(4) 007620 013737 001260 007774 MOV RCSR,TEMP
10262 007626 LET TEMP := TEMP + #24
(7) 007626 062737 000024 007774 ADD #24,TEMP
10263 :ONE INTR IS ALL WE WANT FROM HERE
10264 007634 LET @TEMP := @TEMP CLR.BY #BIT6
(7) 007634 042777 000100 000132 BIC #BIT6,@TEMP
10265 007642 ENDSRV
(1) 007642 000002 RTI
10266
10267 007644 BGNSRV RCVR3SRV
(1) 007644 RCVR3SRV:
10268 :PUT INTR IDENTIFIER IN INTRTABLE
10269 007644 LET (R4)+ := RCH3
(4) 007644 013724 007732 MOV RCH3,(R4)+
10270 :GENERATE CSR ADDRESS
10271 007650 LET TEMP := RCSR
(4) 007650 013737 001260 007774 MOV RCSR,TEMP
10272 007656 LET TEMP := TEMP + #30
```

CNDLAAO DLV11-J TEST MACY11 30(1046) 23-DEC-82 13:56 PAGE 181-2^{K 6}
 CNDLAA.P11 23-DEC-82 13:55 T21 TEST THAT CHANNELS INTR AT ASSIGNED PRIORITY SEQ 0075

| | | | | | | | | | |
|-------|------------|--------|--------|--------|-----|-------------|---------|----------|------------------------------------|
| 10273 | (7) 007656 | 062737 | 000030 | 007774 | ADD | #30,TEMP | | | |
| 10274 | (7) 007664 | 042777 | 000100 | 000102 | BIC | #BIT6,@TEMP | | | :ONE INTR IS ALL WE WANT FROM HERE |
| 10275 | (7) 007672 | 000002 | | | RTI | | ENDSRV | | LET @TEMP := @TEMP CLR.BY #BIT6 |
| 10276 | (1) 007672 | | | | | | | | |
| 10277 | (1) 007674 | | | | | | BGNSRV | XMIT3SRV | |
| 10278 | | | | | | | | | :PUT INTR IDENTIFIER IN INTRTABLE |
| 10279 | (4) 007674 | 013724 | 007742 | | MOV | TCH3,(R4)+ | | | LET (R4)+ := TCH3 |
| 10280 | | | | | | | | | :GENERATE CSR ADDRESS |
| 10281 | (4) 007700 | 013737 | 001260 | 007774 | MOV | RCSR,TEMP | | | LET TEMP := RCSR |
| 10282 | (7) 007706 | 062737 | 000034 | 007774 | ADD | #34,TEMP | | | LET TEMP := TEMP + #34 |
| 10283 | | | | | | | | | :ONE INTR IS ALL WE WANT FROM HERE |
| 10284 | (7) 007714 | 042777 | 000100 | 000052 | BIC | #BIT6,@TEMP | | | LET @TEMP := @TEMP CLR.BY #BIT6 |
| 10285 | (1) 007722 | 000002 | | | RTI | | ENDSRV | | |
| 10286 | | | | | | | SRVEND: | | |

10288
10289
10290
10291
10292
10293
10294
10295
10296
10297 007724 000000
10298 007726 000001
10299 007730 000002
10300 007732 000003
10301 007734 000010
10302 007736 000011
10303 007740 000012
10304 007742 000013
10305
10306
10307
10308
10309
10310
10311 007744 000010
10312 007764
10313
10314 007764 000000
10315 007766 000000
10316 007770 000000
10317 007772 000000
10318 007774 000000
10319

***CHANNEL IDENTIFIER TABLE**
:THE ENTRIES FROM THIS
:TABLE ARE PLACED IN THE INTRTABLE
:IN THE ORDER THAT THE INTR'S
:OCCUR. (EXCLUDING NON-ACTIVE CH'S)

RCH0: .WORD 0
RCH1: .WORD 1
RCH2: .WORD 2
RCH3: .WORD 3
TCH0: .WORD 10
TCH1: .WORD 11
TCH2: .WORD 12
TCH3: .WORD 13

:THIS TABLE WILL CONTAIN ENTRIES
:REPRESENTING EACH INTR IN THE ORDER
:THAT IT OCCURED
: S/B IN THE ABOVE ORDER

INTRTABLE: .BLKW 8.
TABEND:

CHMASK: .WORD 0 ;BITS 3-0 INDICATE WHICH CH'S ARE ACTIVE FOR THIS TEST
CHCNT: .WORD 0 ;CHANNEL COUNTER
PRIERR: .WORD 0 ;ERROR WORD, CONTAINS NO. OF INTR OUT OF ORDER
TCNT: .WORD 0 ;TABLE PASS CTR
TEMP: .WORD 0 ;FOR SERVICE ROUTINES

10321
 10322
 10338
 (3)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (4)
 (3)
 (2) 007776 000004
 (2)
 (1) 010000 012737 000001 001160
 (2) 010006 012737 000022 001200
 10343
 10344 010014
 (4) 010014 005037 012366
 10345
 10346
 10347
 10348 010020
 (4) 010020 012700 011360
 10349 010024
 (3) 010024
 10350 010024
 (4) 010024 005020
 10351 010026
 (3) 010026 020027 012360
 (6) 010032 001374
 10352
 10353
 10354 010034
 (4) 010034 012701 010660
 10355 010040
 (4) 010040 013700 001256
 10356 010044
 (3) 010044
 10357 010044
 (4) 010044 010120
 10358 010046
 (4) 010046 012720 000300
 10359 010052
 (7) 010052 062701 000054
 10360 010056
 (4) 010056 010120
 10361 010060
 (4) 010060 012720 000300
 10362 010064
 (7) 010064 062701 000044
 10363 010070
 (3) 010070 020127 011360
 (6) 010074 001363

```

*****
:TEST 22      TEST DATA TRANSFERS WITH ALL ACTIVE LINES INTERRUPTING
:             IN THIS WE'LL ENABLE INTERRUPTS ON ALL CHANNELS
:             FOR THIS DLV11-J. THEN WE'LL XMIT AN INCREMENTING
:             DATA PATERN VIA INTERRUPTS AND RECORD THE RECEIVER
:             INTR. IN THE RECEIVER STATUS TABLE.
:             NOTE: DOUBLE BUFFERING CANNOT BE TESTED AT ITS MAX SPEED
:             BECAUSE OF APT CONSIDERATIONS. I.E. APT SENDS
:             'BREAKS' WHICH CAUSE OVERRUN ERRORS. THEREFORE
:             THE XMIT IE IS NOT ENABLED AGAIN UNTIL THE RECVR
:             HAS OBTAINED THE PREVIOUS WORD.
*****
TST22: SCOPE
          MOV      #1,$TIMES      ;;DO 1 ITERATION
          MOV      #22,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
          ;;CLR ERROR WORD,
          LET ERWRD := #0

          ;;CLEAR OUT RECEIVER TABLES
          LET RO := #CHOTAB
          REPEAT
            LET (RO)+ := #0
          UNTIL RO EQ #STATEND

          ;;SET UP ALL VECTORS
          LET R1 := #ROSRV
          LET RO := DLVEC
          REPEAT
            LET (RO)+ := R1
            LET (RO)+ := #PR6
            LET R1 := R1 + #54
            LET (RO)+ := R1
            LET (RO)+ := #PR6
            LET R1 := R1 + #44
          UNTIL R1 EQ #SEREND

          $176:
            CLR      (RO)+
            CMP      RO,#STATEND
            BNE     $176

          $177:
            MOV      R1,(RO)+
            MOV      #PR6,(RO)+
            ADD      #54,R1
            MOV      R1,(RO)+
            MOV      #PR6,(RO)+
            ADD      #44,R1
            CMP      R1,#SEREND
            BNE     $177
  
```

```

10364                                ;SET PRIORITY TO 6
10365 010076                        SETPRI #PR6
10366
10367                                ;ENABLE INTR'S ON ALL ACTIVE LINES
10368                                ;THIS MODULE
10369 010110                        LET RO := DLADD
   (4) 010110 013700 001254        MOV DLADD,RO
10370                                ;INITIALIZE
10371 010114                        LET CHMSK := MASK
   (4) 010114 013737 014162 012360  MOV MASK,CHMSK
10372 010122                        LET CHCTR := #0
   (4) 010122 005037 012362        CLR CHCTR
10373                                ;ACTIVE CHANNEL CTR
10374 010126                        LET ACTCH := #0
   (4) 010126 005037 012364        CLR ACTCH
10375 010132                        REPEAT
   (3) 010132
10376 010132                        IF #BIT0 SETIN CHMSK THEN
   (6) 010132 032737 000001 012360  BIT #BIT0,CHMSK
   (9) 010140 001413                BEQ $201
10377                                ;# OF ACTIVE CH
10378 010142                        LET ACTCH := ACTCH + #1
   (7) 010142 005237 012364        INC ACTCH
10379                                ;SET RCSR IE
10380 010146                        LET (RO) := (RO) SET.BY #BIT6
   (7) 010146 052710 000100        BIS #BIT6,(RO)
10381 010152                        LET RO := RO + #4
   (7) 010152 062700 000004        ADD #4,RO
10382                                ;SET XCSR IE
10383 010156                        LET (RO) := (RO) SET.BY #BIT6
   (7) 010156 052710 000100        BIS #BIT6,(RO)
10384 010162                        LET RO := RO + #4
   (7) 010162 062700 000004        ADD #4,RO
10385 010166                        ELSE
   (4) 010166 000402                BR $202
   (3) 010170
10386 010170                        LET RO := RO + #10
   (7) 010170 062700 000010        ADD #10,RO
10387 010174                        ENDIF
   (4) 010174
10388 010174                        LET CHMSK := CHMSK SHIFT -1
   (8) 010174 006237 012360        ASR CHMSK
10389
10390
10391 010200                        LET CHCTR := CHCTR + #1
   (7) 010200 005237 012362        INC CHCTR
10392 010204                        IF CHCTR EQ #3 THEN
   (6) 010204 023727 012362 000003  CMP CHCTR,#3
   (9) 010212 001007                BNE $203
10393 010214                        CALL TSTCON
   (3) 010214 004737 013530        JSR PC,TSTCON
10394 010220                        IF CONSOLE NE #0 THEN
   (6) 010220 005737 013604        TST CONSOLE
   (9) 010224 001402                BEQ $204
10395 010226                        LET CHCTR := CHCTR + #1
   (7) 010226 005237 012362        INC CHCTR

```



```

10396 010232                                ENDIF
      (4) 010232                                $204:                                ENDIF
10397 010232                                $203:                                UNTIL CHCTR EQ #4
      (4) 010232
10398 010232                                CMP      CHCTR,#4
      (3) 010232 023727 012362 000004        BNE      $200
      (6) 010240 001334
10399
10400                                ;INIT BUFFER POINTERS FOR SERVICE ROUTINES
10401 010242                                LET R0 := #CH0TAB
      (4) 010242 012700 011360                MOV      #CH0TAB,R0
10402 010246                                LET R1 := #CH1TAB
      (4) 010246 012701 011560                MOV      #CH1TAB,R1
10403 010252                                LET R2 := #CH2TAB
      (4) 010252 012702 011760                MOV      #CH2TAB,R2
10404 010256                                LET R3 := #CH3TAB
      (4) 010256 012703 012160                MOV      #CH3TAB,R3
10405
10406                                ;FILL XBUF WORDS
10407                                ;FROM TMP0 THRU TMP3E
10408                                ;START WORD CH 0 ALWAYS 0
10409 010262                                LET TMP0 := #0
      (4) 010262 005037 012372                CLR      TMP0
10410                                ;OFFSET TO INDEX
10411 010266                                LET R4 := #0
      (4) 010266 005004                CLR      R4
10412                                ;PTR TO TABLE TO BE FILLED
10413 010270                                LET R5 := #TMPOE
      (4) 010270 012705 012374                MOV      #TMPOE,R5
10414 010274                                REPEAT
      (3) 010274                                IF #BIT0 NOTSETIN $USWR THEN
10415 010274                                BIT      #BIT0,$USWR
      (6) 010274 032737 000001 001220        BNE      $206
      (9) 010302 001005
10416 010304                                LET (R5)+ := TABL7(R4)
      (4) 010304 016425 012426                MOV      TABL7(R4),(R5)+
10417 010310                                LET (R5)+ := TABL7(R4)
      (4) 010310 016425 012426                MOV      TABL7(R4),(R5)+
10418 010314                                ELSE
      (4) 010314 000404                BR       $207
      (3) 010316                                $206:
10419 010316                                LET (R5)+ := TABL8(R4)
      (4) 010316 016425 012436                MOV      TABL8(R4),(R5)+
10420 010322                                LET (R5)+ := TABL8(R4)
      (4) 010322 016425 012436                MOV      TABL8(R4),(R5)+
10421 010326                                ENDIF
      (4) 010326                                $207:
10422 010326                                LET R4 := R4 + #2
      (7) 010326 062704 000002                ADD      #2,R4
10423 010332                                UNTIL R4 EQ #10
      (3) 010332 020427 000010                CMP      R4,#10
      (6) 010336 001356                BNE      $205
10424                                ;INIT TMP4
10425                                ;WILL BE DECR BY XMIT SERVICE ROUTINES TILL 0
10426 010340                                LET TMP4 := ACTCH
      (4) 010340 013737 012364 012414        MOV      ACTCH,TMP4
    
```


| | | | | | | |
|-------|--------|--------|---------------|------|---------------|---------------------------------|
| 10461 | 010474 | | | | | LET R1 := TMPO(R0) |
| (4) | 010474 | 016001 | 012372 | MOV | TMPO(R0),R1 | LET R0 := R0 + #2 |
| 10462 | 010500 | | | ADD | #2,R0 | ;GET MAX WORD FOR THAT CH. |
| (7) | 010500 | 062700 | 000002 | | | LET R3 := TMPO(R0) |
| 10463 | | | | | | REPEAT |
| 10464 | 010504 | | | MOV | TMPO(R0),R3 | IF #BIT15 SETIN (R2) THEN |
| (4) | 010504 | 016003 | 012372 | | | LET ERWRD := ERWRD SET.BY #BIT8 |
| 10465 | | | | | | ENDIF |
| 10466 | 010510 | | | | | ;CHECK FOR DATA COMPARE ERROR |
| (3) | 010510 | | | | | IFB (R2) NE R1 THEN |
| 10467 | 010510 | | | BIT | #BIT15,(R2) | LET ERWRD := ERWRD + #1 |
| (6) | 010510 | 032712 | 100000 | BEQ | \$215 | ENDIF |
| (9) | 010514 | 001403 | | | | |
| 10468 | 010516 | | | BIS | #BIT8,ERWRD | |
| (7) | 010516 | 052737 | 000400 012366 | | | |
| 10469 | 010524 | | | | | |
| (4) | 010524 | | | | | |
| 10470 | | | | | | |
| 10471 | 010524 | | | CMPB | (R2),R1 | |
| (6) | 010524 | 121201 | | BEQ | \$216 | |
| (9) | 010526 | 001402 | | | | |
| 10472 | 010530 | | | INC | ERWRD | |
| (7) | 010530 | 005237 | 012366 | | | |
| 10473 | 010534 | | | | | |
| (4) | 010534 | | | | | |
| 10474 | | | | | | |
| 10475 | 010534 | | | INC | R1 | |
| (7) | 010534 | 005201 | | | | |
| 10476 | | | | | | |
| 10477 | 010536 | | | ADD | #2,R2 | |
| (7) | 010536 | 062702 | 000002 | | | |
| 10478 | | | | | | |
| 10479 | 010542 | | | | | UNTIL R1 EQ R3 OR ERWRD NE #0 |
| (4) | 010542 | 020103 | | CMP | R1,R3 | |
| (6) | 010544 | 001403 | | BEQ | \$217 | |
| (4) | 010546 | 005737 | 012366 | TST | ERWRD | |
| (7) | 010552 | 001756 | | BEQ | \$214 | |
| (4) | 010554 | | | | | |
| 10480 | 010554 | | | | | ENDIF |
| (4) | 010554 | | | | | LET CHCTR := CHCTR + #1 |
| 10481 | 010554 | | | INC | CHCTR | LET CHMSK := CHMSK SHIFT -1 |
| (7) | 010554 | 005237 | 012362 | | | IF CHCTR EQ #3 THEN |
| 10482 | 010560 | | | ASR | CHMSK | CALL TSTCON |
| (8) | 010560 | 006237 | 012360 | | | IF CONSOLE EQ #TRUE THEN |
| 10483 | 010564 | | | | | |
| (6) | 010564 | 023727 | 012362 000003 | CMP | CHCTR,#3 | |
| (9) | 010572 | 001010 | | BNE | \$220 | |
| 10484 | 010574 | | | JSR | PC,TSTCON | |
| (3) | 010574 | 004737 | 013530 | | | |
| 10485 | 010600 | | | | | |
| (6) | 010600 | 023727 | 013604 000001 | CMP | CONSOLE,#TRUE | |
| (9) | 010606 | 001002 | | BNE | \$221 | |
| 10486 | | | | | | |
| 10487 | 010610 | | | | | ;EXIT |
| (7) | 010610 | 005237 | 012362 | INC | CHCTR | LET CHCTR := CHCTR + #1 |
| 10488 | 010614 | | | | | ENDIF |


```

(4) 010614 $221:
10489 010614 $220:
(4) 010614 $220:
10490 010614 UNTIL CHCTR EQ #4 OR ERWRD NE #0
(4) 010614 023727 012362 000004 CMP CHCTR,#4
(6) 010622 001403 BEQ $222
(4) 010624 005737 012366 TST ERWRD
(7) 010630 001707 BEQ $212
(4) 010632 $222:
10491
10492 010632 IF #BIT8 SET IN ERWRD THEN
(6) 010632 032737 000400 012366 BIT #BIT8,ERWRD
(9) 010640 001401 BEQ $223
10493 :ERROR FLAG UP AFTER TRANSFER
10494 010642 ERRHRD 270
(1) 010642 104270 ERROR 270
10495 010644 ENDIF
(4) 010644 $223: IFB ERWRD NE #0 THEN
10496 010644 TSTB ERWRD
(6) 010644 105737 012366 BEQ $224
(9) 010650 001401
10497 :DATA COMPARE ERROR
10498 010652 ERRHRD 271
(1) 010652 104271 ERROR 271
10499 010654 ENDIF
(4) 010654 $224:
10500
10501
10502 010654 000137 006502 JMP TST20 ;EXIT TEST
  
```

```
10504
10505
10510
10511
10512
10517
10518 010660
(1) 010660
10519 010660
(4) 010660 013737 001260 012370
10520 010666
(7) 010666 062737 000002 012370
10521 010674
(4) 010674 017720 001470
10522 010700
(6) 010700 023737 012372 012374
(9) 010706 001407
10523
10524 010710
(7) 010710 062737 000002 012370
10525
10526 010716
(7) 010716 052777 000100 001444
10527 010724
(4) 010724 000402
(3) 010726
10528
10529 010726
(7) 010726 005337 012414
10530 010732
(4) 010732
10531 010732
(1) 010732 000002
10532
10533 010734
(1) 010734
10534 010734
(4) 010734 013737 001260 012370
10535 010742
(7) 010742 062737 000006 012370
10536 010750
(4) 010750 013777 012372 001412
10537 010756
(7) 010756 005237 012372
10538
10539 010762
(7) 010762 162737 000002 012370
10540
10541
10542 010770
(7) 010770 042777 000100 001372
10543 010776
(1) 010776 000002
10544
10545 011000
(1) 011000
```

* START OF SERVICE ROUTINES *

```
ROSRV:
MOV RCSR,TMP
ADD #2,TMP
MOV @TMP,(RO)+
CMP TMPO,TMPOE
BEQ $225
ADD #2,TMP
BIS #BIT6,@TMP
BR $226
$225:
DEC TMP4
$226:
RTI
XOSRV:
MOV RCSR,TMP
ADD #6,TMP
MOV TMPO,@TMP
INC TMPO
SUB #2,TMP
RTI
R1SRV:
```

```
BGNSRV ROSRV
LET TMP := RCSR
LET TMP := TMP + #2
LET (RO)+ := @TMP
IF TMPO NE TMPOE THEN
;GO TO XCSR
LET TMP := TMP + #2
;ENABLE XMIT INTERR
LET @TMP := @TMP SET.BY #BIT6
ELSE
;ALL DONE
LET TMP4 := TMP4 - #1
ENDIF
ENDSRV
BGNSRV XOSRV
LET TMP := RCSR
LET TMP := TMP + #6
LET @TMP := TMPO
LET TMPO := TMPO + #1
;GO BACK TO XCSR
LET TMP := TMP - #2
;DISABLE XMIT INTERRUPTS,
;NOT EXER DOUBLE BUFFERING
LET @TMP := @TMP CLR.BY #BIT6
ENDSRV
BGNSRV R1SRV
```

```

10546 011000          MOV      RCSR,TMP          LET TMP := RCSR
(4) 011000 013737 001260 012370
10547 011006          ADD      #12,TMP          LET TMP := TMP + #12
(7) 011006 062737 000012 012370
10548 011014          MOV      @TMP,(R1)+        LET (R1)+ := @TMP
(4) 011014 017721 001350
10549 011020          CMP      TMP1,TMP1E        IF TMP1 NE TMP1E THEN
(6) 011020 023737 012376 012400
(9) 011026 001407          BEQ      $227
10550
10551 011030          ADD      #2,TMP          ;GO TO XCSR
(7) 011030 062737 000002 012370          LET TMP := TMP + #2
10552
10553 011036          BIS      #BIT6,@TMP        ;ENABLE XMIT INTERR
(7) 011036 052777 000100 001324          LET @TMP := @TMP SET.BY #BIT6
10554 011044          BR       $230          ELSE
(4) 011044 000402
(3) 011046          $227:
10555
10556 011046          DEC      TMP4          ;ALL DONE
(7) 011046 005337 012414          LET TMP4 := TMP4 - #1
10557 011052          $230:
(4) 011052
10558 011052          RTI          ENDSRV
(1) 011052 000002
10559
10560 011054          X1SRV:          BGNSRV X1SRV
(1) 011054
10561 011054          MOV      RCSR,TMP          LET TMP := RCSR
(4) 011054 013737 001260 012370
10562 011062          ADD      #16,TMP          LET TMP := TMP + #16
(7) 011062 062737 000016 012370
10563 011070          MOV      TMP1,@TMP        LET @TMP := TMP1
(4) 011070 013777 012376 001272
10564 011076          INC      TMP1          LET TMP1 := TMP1 + #1
(7) 011076 005237 012376
10565
10566 011102          SUB      #2,TMP          ;GO BACK TO XCSR
(7) 011102 162737 000002 012370          LET TMP := TMP - #2
10567
10568          ;DISABLE XMIT INTERRUPTS,
10569 011110          BIC      #BIT6,@TMP        ;NOT EXER DOUBLE BUFF
(7) 011110 042777 000100 001252          LET @TMP := @TMP CLR.BY #BIT6
10570 011116          RTI          ENDSRV
(1) 011116 000002
10571
10572 011120          R2SRV:          BGNSRV R2SRV
(1) 011120
10573 011120          MOV      RCSR,TMP          LET TMP := RCSR
(4) 011120 013737 001260 012370
10574 011126          ADD      #22,TMP          LET TMP := TMP + #22
(7) 011126 062737 000022 012370
10575 011134          MOV      @TMP,(R2)+        LET (R2)+ := @TMP
(4) 011134 017722 001230
10576 011140          CMP      TMP2,TMP2E        IF TMP2 NE TMP2E THEN
(6) 011140 023737 012402 012404

```



```

(9) 011146 001407          BEQ      $231
10577                                ;GO TO XCSR
10578 011150                ADD      #2,TMP      LET TMP := TMP + #2
(7) 011150 062737 000002 012370          ;ENABLE XMIT INTERR
10579                                LET @TMP := @TMP SET.BY #BIT6
10580 011156                BIS      #BIT6,@TMP
(7) 011156 052777 000100 001204          ELSE
10581 011164                BR       $232
(4) 011164 000402          $231:
(3) 011166
10582                                ;ALL DONE
10583 011166                DEC      TMP4      LET TMP4 := TMP4 - #1
(7) 011166 005337 012414          ENDIF
10584 011172                $232:
(4) 011172
10585 011172                ENDSRV
(1) 011172 000002          RTI
10586
10587 011174                X2SRV:
(1) 011174
10588 011174                MOV      RCSR,TMP      BGNSRV X2SRV
(4) 011174 013737 001260 012370          LET TMP := RCSR
10589 011202                ADD      #26,TMP      LET TMP := TMP + #26
(7) 011202 062737 000026 012370          LET @TMP := TMP2
10590 011210                MOV      TMP2,@TMP      LET TMP2 := TMP2 + #1
(4) 011210 013777 012402 001152          INC      TMP2
10591 011216                SUB      #2,TMP      ;GO BACK TO XCSR
(7) 011216 005237 012402          LET TMP := TMP - #2
10592                                ;DISABLE XMIT INTERRUPTS,
10593 011222                SUB      #2,TMP      ;NOT EXER DOUBLE BUFF.
(7) 011222 162737 000002 012370          LET @TMP := @TMP CLR.BY #BIT6
10594                                ENDSRV
10595
10596 011230                BIC      #BIT6,@TMP
(7) 011230 042777 000100 001132          RTI
10597 011236                R3SRV:
(1) 011236 000002
10598
10599 011240                MOV      RCSR,TMP      BGNSRV R3SRV
(1) 011240
10600 011240                ADD      #32,TMP      LET TMP := RCSR
(4) 011240 013737 001260 012370          LET TMP := TMP + #32
10601 011246                MOV      @TMP,(R3)+      LET (R3)+ := @TMP
(7) 011246 062737 000032 012370          IF TMP3 NE TMP3E THEN
10602 011254                CMP      TMP3,TMP3E
(4) 011254 017723 001110          BEQ      $233
10603 011260                ;GO TO XCSR
(6) 011260 023737 012406 012410          LET TMP := TMP + #2
(9) 011266 001407          ;ENABLE XMIT INTERR
10604                                LET @TMP := @TMP SET.BY #BIT6
10605 011270                ADD      #2,TMP      ELSE
(7) 011270 062737 000002 012370
10606                                ;GO TO XCSR
10607 011276                BIS      #BIT6,@TMP      LET TMP := TMP + #2
(7) 011276 052777 000100 001064          ;ENABLE XMIT INTERR
10608 011304                LET @TMP := @TMP SET.BY #BIT6

```

```

(4) 011304 000402
(3) 011306
10609
10610 011306
(7) 011306 005337 012414
10611 011312
(4) 011312
10612 011312
(1) 011312 000002
10613
10614 011314
(1) 011314
10615 011314
(4) 011314 013737 001260 012370
10616 011322
(7) 011322 062737 000036 012370
10617 011330
(4) 011330 013777 012406 001032
10618 011336
(7) 011336 005237 012406
10619
10620 011342
(7) 011342 162737 000002 012370
10621
10622
10623 011350
(7) 011350 042777 000100 001012
10624 011356
(1) 011356 000002
10625 011360
    
```

```

BR $234
$233:
DEC TMP4
$234:
RTI
X3SRV:
MOV RCSR,TMP
ADD #36,TMP
MOV TMP3,@TMP
INC TMP3
SUB #2,TMP
BIC #BIT6,@TMP
RTI
SEREND:
    
```

```

:ALL DONE
LET TMP4 := TMP4 - #1
ENDIF
ENDSRV
BGNSRV X3SRV
LET TMP := RCSR
LET TMP := TMP + #36
LET @TMP := TMP3
LET TMP3 := TMP3 + #1
:GO BACK TO XCSR
LET TMP := TMP - #2
:DISABLE XMIT INTERRUPTS,
:NOT EXER DOUBLE BUFF
LET @TMP := @TMP CLR.BY #BIT6
ENDSRV
    
```

10627
 10628
 10629
 10630
 10631 011360 000100
 10632 011560 000100
 10633 011760 000100
 10634 012160 000100
 10635 012360
 10636 012360 000000
 10637 012362 000000
 10638 012364 000000
 10639 012366 000000
 10640
 10641
 10642 012370 000000
 10643
 10644 012372 000000
 10645 012374 000000
 10646 012376 000000
 10647 012400 000000
 10648 012402 000000
 10649 012404 000000
 10650 012406 000000
 10651 012410 000002
 10652
 10653 012414 000000
 10654
 10655 012416 011360
 10656 012420 011560
 10657 012422 011760
 10658 012424 012160
 10659
 10660 012426 000040
 10661 012430 000100
 10662 012432 000140
 10663 012434 000200
 10664
 10665 012436 000100
 10666 012440 000200
 10667 012442 000300
 10668 012444 000400
 10669
 10670

;** RECEIVER STATUS TABLES **

CH0TAB: .BLKW 100
 CH1TAB: .BLKW 100
 CH2TAB: .BLKW 100
 CH3TAB: .BLKW 100
 STATEND:
 CHMSK: .WORD 0
 CHCTR: .WORD 0
 ACTCH: .WORD 0
 ERWRD: .WORD 0

 TMP: .WORD 0

 TMP0: 0
 TMP0E: 0
 TMP1: 0
 TMP1E: 0
 TMP2: 0
 TMP2E: 0
 TMP3: 0
 TMP3E: .BLKW 2

 TMP4: .WORD

 TABL6: CH0TAB
 CH1TAB
 CH2TAB
 CH3TAB

 TABL7: 40
 100
 140
 200

 TABL8: 100
 200
 300
 400

:7 BIT WDS
 : 0- 37
 : 40- 77
 :100-137
 :140-177

 8 BIT WDS
 0- 77
 100-177
 200-277
 300-377

 :MASK OF ACTIVE CHANNELS
 :CHANNEL CTR
 :ACTIVE CHANNEL CTR
 :ERROR WORD, LOW BYTE IS NO. OF DATA ERRORS
 :HIGH BYTE INDICATES ERROR FLAG DETECTED

 :TEMP STORAGE FOR SERVICE ROUTINES

 :START WORD FOR CH 0
 :END WORD FOR CH 0
 :START 1
 :END 1
 :START 2
 :END 2
 :START 3
 :END 3

 :7 BIT WORD TABLE

 :8 BIT WORD TABLE

10672
 10673
 10678
 10679
 10680
 10681
 10682
 10683
 10684
 10685
 10686
 10687
 10688
 10689
 10690
 10691
 10692
 10693
 10694
 10695
 10696
 10697
 10698
 10699

```

*****
* ROUTINE:TIMER
* THIS ROUTINE IS USED TO TEST THE STATUS OF ANY BIT
* IN ANY REGISTER.
* INPUTS:
*   HOWLONG   THE MAXIMUM AMOUNT OF TIME TO SPEND IN
*             THIS ROUTINE.
*   WHICHBIT  A MASK WITH THE BIT(S) SET THAT ARE
*             TO BE CHECKED.
*   REG       A POINTER TO THE REGISTER TO BE CHECKED
*   SETCLR    THE DESIRED RESULTS
*             EITHER #SET OR #CLEAR
* OUTPUT:
*   THE 'C' BIT IS SET TO INDICATE AN ERROR
*   BUT IT IS TESTED BY THE IF.ERROR STATEMENT
*
* NOTE:: THE USE OF (R5) IS PART OF THE LINKAGE
*        MECHANISM BETWEEN THE CALLER AND THE CALLED
*****
  
```

10704 012446
 (2) 012446
 10705 012446 016537 000004 012606
 (4) 012446 016537 000000 012610
 10706 012454
 (4) 012454 016537 000000 012610
 10707 012462
 (4) 012462 105037 012612
 10708
 10709
 10710
 10711
 10712 012466
 (3) 012466
 10713
 10714 012466 036577 000002 000112
 (6) 012466 001003
 (9) 012474
 10715 012476 105037 012613
 (4) 012476
 10716 012502 000403
 (4) 012502
 (3) 012504
 10717 012504 112737 177777 012613
 (4) 012504
 10718 012512
 (4) 012512
 10719
 10720
 10721 012512 123765 012613 000006
 (6) 012512 001003
 (9) 012520

```

ROUTINE TIMER <HOWLONG,WHICHBIT,REG,SETCLR>
TIMER:
MOV REG(R5),REGSAV          LET REGSAV := REG(R5) ; GET POINTER TO REGIST
MOV HOWLONG(R5),TIMSAV     LET TIMSAV := HOWLONG(R5) ; SAVE HOWLONG FOR
CLRB FLAG                  LET FLAG :B= #0 ; INITIALIZE THE EXIT FLAG
; START OF AN INFINITE LOOP
LOOP
; TEST TO SEE IF WHICHBIT IS SET
IF WHICHBIT(R5) NOTSETIN @REGSAV THEN
LET HOLDSC :B= #0
ELSE
LET HOLDSC :B= #SET ; REMEMBER THIS
ENDIF
; NOW SEE IF THAT WAS WHAT WE WANTED
IFB HOLDSC EQ SETCLR(R5) THEN
  
```


10750
 10751
 10756
 10757
 10758
 10759
 10760
 10761
 10762
 10763
 10768 012620
 (2) 012620
 10769 012620
 10770 012626
 (4) 012626 016501 000000
 10771 012632
 (4) 012632 012702 000001
 (5) 012636 000402
 (4) 012640
 (8) 012640 062702 000001
 (5) 012644
 (5) 012644 020201
 (7) 012646 101010
 10772 012650
 (4) 012650 005003
 (5) 012652 000401
 (4) 012654
 (8) 012654 005203
 (5) 012656
 (5) 012656 020327 000100
 (7) 012662 003001
 10773 012664
 (4) 012664 000773
 (3) 012666
 10774 012666
 (4) 012666 000764
 (3) 012670
 10775 012670
 10776 012676
 (3) 012676
 (3) 012676
 (2) 012676 000207

```

*****
* ROUTINE:WAIT
* THIS ROUTINE IS USED TO DELAY EXECUTION OF THE
* MAIN PROGRAM FOR A SPECIFIED AMOUNT OF TIME.
* THIS IS ACCOMPLISHED BY INCREMENTING A
* REGISTER UP TO A LIMIT. THE INNER LOOP IS SET
* TO APPROXIMATE 1 MILLI SEC.
*****
ROUTINE WAIT <TIME>
  
```

```

WAIT:
      PUSH <R1,R2,R3>
      LET R1 := TIME(R5)
      MOV TIME(R5),R1
      INCRU R2 FROM #1 TO R1 BY #1
      MOV #1,R2
      BR $247
$250: ADD #01,R2
$247: CMP R2,R1
      BHI $251
      INCR R3 FROM #0 TO #100 BY #1
      CLR R3
      BR $252
$253: INC R3
$252: CMP R3,#100
      BGT $254
      ENDINC
$254: BR $253
      ENDINC
$251: BR $250
      POP <R3,R2,R1>
      ENDRTN
$245:
$246: RTS PC
  
```



```

10778
10779
10780
10781          .SBTTL  ROUTINE TO SIZE THE BUS
10782
10783 012700 005737 001252      SIZE:  TST      $DEVN      ;BR IF INITIALLY SET
10784 012704 001060              BNE      4$          ;BYPASS AUTO-SIZING
10785
10786 012706 012737 013610 000004 1$:  MOV      #INTSRV,ERRVEC
10787 012714 012737 000300 000006      MOV      #PR6,ERRVEC+2
10788
10789 012722 013737 001250 013514      MOV      $BASE,TMP6
10790 012730 012737 000001 013512      MOV      #1,SFTREG
10791
10792 012736 005037 013616      2$:  CLR      INTFLAG
10793 012742 005777 000546          TST      @TMP6      ;SEE IF THERE
10794 012746 005737 013616          TST      INTFLAG   ;GOT INTERRUPT?
10795 012752 001003              BNE      3$          ;BR IF YES (NO DEVICE)
10796 012754 053737 013512 001252      BIS      SFTREG,$DEVN ;ELSE THERE, SET MAP
10797
10798 012762 062737 000010 013514 3$:  ADD      #10,TMP6   ;GO TO NEXT CHANNEL
10799 012770 006337 013512          ASL      SFTREG
10800 012774 001360              BNE      2$          ;BR IF MORE TO GO
10801
10802 012776 012737 000006 000004      MOV      #6,ERRVEC  ;RESTORE
10803 013004 005037 000006              CLR      ERRVEC+2
10804
10805 013010 005737 001252          TST      $DEVN      ;ANYONE THERE?
10806 013014 001014              BNE      4$          ;BR IF YES
10807 013016              TYPTXT  <<CRLF>!NO DEV TO TEST!>
10808 013044 000777              BR      .           ;LOOP HERE TILL PROBLEM FIXED
10809                                ;& RESTART AT 200
10810 013046 105737 001221      4$:  TSTB    $USWR+1   ;SEE IF ANY CONSOLE
10811 013052 001434              BEQ      6$          ;BR IF NONE
10812
10813 013054 010046              MOV      RO,-(SP)   ;SAVE RO
10814 013056 113700 001221      MOVB    $USWR+1,RO ;GET CONSOLE DEVICE X2
10815 013062 020027 000010      CMP     RO,#10
10816 013066 003005              BGT     5$          ;BR IF TOO HI
10817 013070 056037 013516 001252      BIS     TABL1-2(RO),$DEVN ;SET CONSOLE BIT ON MAP
10818 013076 012600              MOV     (SP)+,RO   ;RESTORE RO
10819 013100 000421              BR      6$
10820
10821 013102      5$:  TYPTXT  <<CRLF>!CONSOLE IN $USWR WRONG!>
10822 013140 000137 014362      JMP     $EOP        ;START ALL OVER
10823                                ;STACK WILL RESET IN 'LOOP'
10824 013144 013737 001252 013502 6$:  MOV     $DEVN,MASK1
10825 013152 013737 001252 013504      MOV     $DEVN,MASK2
10826 013160 013737 001252 013506      MOV     $DEVN,MASK3
10827 013166 013737 001252 013510      MOV     $DEVN,MASK4
10828
10829 013174 042737 177760 013502      BIC     #^C<17>,MASK1 ;SAVE ONLY MODULE 1 CHANNELS
10830 013202 042737 177417 013504          BIC     #^C<360>,MASK2 ;
10831 013210 042737 170377 013506          BIC     #^C<7400>,MASK3 ;
10832 013216 042737 007777 013510          BIC     #^C<170000>,MASK4 ;
10833

```

```

10834 013224 006237 013504      ASR  MASK2
10835 013230 006237 013504      ASR  MASK2
10836 013234 006237 013504      ASR  MASK2
10837 013240 006237 013504      ASR  MASK2
10838                                     ;RIGHT JUSTIFY MODULE 2 CHANNELS
10839 013244 000337 013506      SWAB  MASK3
10840                                     ;
10841 013250 000337 013510      SWAB  MASK4
10842 013254 006237 013510      ASR  MASK4
10843 013260 006237 013510      ASR  MASK4
10844 013264 006237 013510      ASR  MASK4
10845 013270 006237 013510      ASR  MASK4
10846                                     ;
10847 013274      TYPTXT  <<CRLF>!WILL TEST:!!<CRLF>>
10848
10849 013320 012700 013502      MOV  #MASK1,RO
10850 013324 012737 000061 013474  MOV  #61,MODNUM      ;INIT MOD #
10851 013332 012737 000060 013500  MOV  #60,CHNUM+2    ;INIT CHANNEL #
10852
10853 013340 012037 013516      7$:  MOV  (RO)+,MSK
10854 013344 001001      BNE  8$              ;BR IF ANY THERE
10855 013346 000441      BR   11$            ;ELSE TRY NEXT MODULE
10856
10857 013350      8$:  TYPTXT  <<CRLF>!MODULE !>
10858 013370 104401 013474      TYPE  ,MODNUM
10859 013374      TYPTXT  <! CHANNEL !>
10860
10861 013416 032737 000001 013516  9$:  BIT  #BIT0,MSK
10862 013424 001402      BEQ  10$            ;BR IF NOT TESTING THAT CHANNEL
10863 013426 104401 013476      TYPE  ,CHNUM      ;ELSE TYPE IT
10864
10865 013432 005237 013500      10$: INC  CHNUM+2      ;BUMP CHANNEL CTR
10866 013436 006237 013516      ASR  MSK
10867 013442 001365      BNE  9$              ;BR IF MORE TO CHECK
10868 013444 012737 000060 013500  MOV  #60,CHNUM+2    ;ELSE RESET CH #
10869
10870 013452 005237 013474      11$: INC  MODNUM      ;BUMP MOD #
10871 013456 023727 013474 000065  CMP  MODNUM,#65    ;ALL DONE?
10872 013464 001325      BNE  7$              ;BR IF NO
10873 013466 104401 001171      TYPE  ,SCRLF
10874 013472 000207      RTS   PC            ;ELSE EXIT
10875
10876 013474 000061      MODNUM: .ASCIZ  /1/
10877 013476 020040 000060      CHNUM:  .ASCIZ  / 0/
10878
10879 013502 000000      MASK1:  0          ;CONTAINS MODULE 1 CHANNELS
10880 013504 000000      MASK2:  0          ;
10881 013506 000000      MASK3:  0          ;
10882 013510 000000      MASK4:  0          ;
10883 013512 000000      SFTREG: 0          ;
10884 013514 000000      TMP6:   0          ;
10885 013516 000000      MSK:    0          ;
10886 013520 000010      TABL1:  10         ;SET $DEVN IF CONSOLE ON MODULE 1
10887 013522 000200      ;
10888 013524 004000      ;
10889 013526 100000      ;

```

```

10890
10891
10892
10893
10894          .SBTTL  ROUTINE TO TEST FOR CONSOLE DEVICE
10895
10896 013530 113737 001221 013606 TSTCON: MOVB  $USWR+1,TMP7          ;GET CONSOLE INFO ONLY
10897 013536 006237 013606          ASR    TMP7              ;RIGHT JUSTIFY
10898 013542 023737 013606 001206      CMP    TMP7,$UNIT        ;SEE IF UNIT UNDER TEST HAS CONSOLE
10899 013550 001403          BEQ    1$                ;BR IF YES
10900 013552 005037 013604          CLR    CONSOLE          ;ELSE CLEAR STALE INFO
10901 013556 000207          RTS    PC
10902
10903 013560 013737 001144 014156 1$:   MOV    $TKS,ADDRESS      ;FORCE TTY ADDRESS
10904 013566 012737 000060 014160      MOV    #TKVEC,VECTOR    ;
10905 013574 012737 000001 013604      MOV    #TRUE,CONSOLE    ;SET FLAG
10906 013602 000207          RTS    PC
10907
10908 013604 000000          CONSOLE: 0              ;FLAG, 1= TESTING CONSOLE DEVICE
10909 013606 000000          TMP7:   0              ;HOLD CONSOLE #
10910
10911
10912          .SBTTL  INTSRV  INTERRUPT SERVICE ROUTINE
10913
10914
10919          ;*****
10920          ; THIS GLOBAL ROUTINE DOES NOTHING BUT INCREMENT
10921          ; 'INTFLAG' EACH TIME IT IS CALLED.  IT ASSUMES
10922          ; THAT THE MAIN CALLING ROUTINE WILL KNOW WHAT
10923          ; TO LOOK FOR.
10924          ;*****
10929 013610          BGNSRV INTSRV          ;GLOBAL INTERRUPT SERVICE ROUTINE
10930 (1) 013610          INTSRV:
10931          ;ADD 1 TO 'INTERRUPT OCCURED' FLAG
10931 013610          INC    INTFLAG          LET INTFLAG := INTFLAG + #1
10932 (7) 013610 005237 013616          ENDSRV
10932 (1) 013614 000002          RTI
10933 013616 000000          INTFLAG: 0
10934
  
```


10936
 10937
 10938
 10939
 10940
 10945
 10946
 10947
 10948
 10949
 10954
 10955
 10956
 10957
 10958
 10959
 10960
 10961
 10962
 10963
 10964
 10965
 10966
 10967
 10968
 10969
 10970
 10971
 10972
 10973
 10974
 10975
 10976
 10977
 10978
 10979
 10980
 10981
 10982
 10983
 10984
 10985
 10986
 10987
 10988
 10989
 10990
 10991
 10992
 10993
 10994
 10995
 10996
 10997
 10998
 10999

.SBTTL ROUTINE TO SET UP FOR NEXT MODULE/CHANNEL ADDRESS

```

:*****
:* THIS ROUTINE CAUSES ADRS TO POINT TO THE
:* ADDRESS OF DLV11-J UNDER TEST, ADRS +2 TO
:* POINT TO THE VECTOR OF THE DLV11-J UNDER TEST.
:*****
  
```

```

CYCLE:  MOV  $BASE,ADDRESS
        MOV  $VECT1,VECTOR

        CMP  $UNIT,#5           ;DONE ALL MODULES?
        BNE  1$                ;BR IF NO
        CLR  PHASE2            ;RE-INIT
        CLR  CHAN
        MOV  #1,$UNIT
        CLR  P1CNT
        JMP  $EOP              ;& EXIT BACK TO 'LOOP'
                                       ;STACK WILL GET RESET THERE.

1$:     MOV  $UNIT,R0
        ASL  R0                ;MULT BY 2 TO INDEX INTO TABLES
        ADD  TABL2-2(R0),ADDRESS ;ADD MODULE OFFSET
        ADD  TABL2-2(R0),VECTOR  ;ADD VECTOR OFFSET
        MOV  @TABL3-2(R0),MASK   ;STORE CHANNELS IN 'MASK'

        CLR  CONSOLE
        TST  MASK              ;ANY CHANNELS TO BE TESTED?
        BNE  2$                ;BR IF YES
        INC  $UNIT             ;ELSE BUMP UNIT #
        JMP  CYCLE             ;TRY AGAIN

2$:     CMP  PHASE2,#TRUE      ;DOING PHASE 2?
        BEQ  6$                ;BR IF YES
        CMP  CHAN,#4           ;ELSE, DONE ALL CHANNELS?
        BNE  4$                ;BR IF NO

        CLR  CHAN              ;ELSE BACK TO 0
        BIT  #WRAP,$USWR       ;DOING DATA WRAP TESTS?
        BNE  3$                ;BR IF YES
        INC  $UNIT             ;ELSE GO TO NEXT UNIT, NO PHASE 2
        JMP  CYCLE             ;& DO IT

3$:     MOV  #TRUE,PHASE2
        CLR  P1CNT
        JMP  CYCLE              ;GO TO PHASE 2 TESTING

4$:     MOV  CHAN,R0
        ASL  R0                ;MULT BY 2 TO INDEX TABLE
        ADD  TABL4(R0),ADDRESS  ;ADD CHANNEL OFFSET
        ADD  TABL4(R0),VECTOR  ;ADD VECTOR OFFSET
        BIT  TABL5(R0),MASK     ;IS THAT CHANNEL TO BE TESTED?
        BEQ  7$                ;BR IF NO
  
```

```

11000 014060 023727 014164 000003      CMP      CHAN,#3      ;IS IT CHAN 3?
11001 014066 001002                      BNE      5$          ;BR IF NO
11002 014070 004737 013530      JSR      PC,TSTCON  ;ELSE SEE IF ITS THE CONSOLE
11003
11004 014074 005237 014164      5$:      INC      CHAN      ;BUMP CHANNEL CTR FOR NEXT TIME
11005 014100 012701 014156      6$:      MOV      #ADDRESS,ADRS
11006 014104 000207                      RTS      PC          ;GO TO PHASE 1 TESTS
11007
11008 014106 005237 014164      7$:      INC      CHAN
11009 014112 000137 013620      JMP      CHAN CYCLE ;TRY AGAIN
11010
11011 014116 000000      TABL2:  0          ;OFFSET FOR MODULE 1
11012 014120 000040                      40          :                2
11013 014122 000100                      100         :                3
11014 014124 000140                      140         :                4
11015
11016 014126 013502      TABL3:  MASK1      ;MODULE ? CHANNEL DATA
11017 014130 013504                      MASK2      :                2
11018 014132 013506                      MASK3      :                3
11019 014134 013510                      MASK4      :                4
11020
11021 014136 000000      TABL4:  0          ;CHANNEL 0 OFFSET
11022 014140 000010                      10         :                1
11023 014142 000020                      20         :                2
11024 014144 000030                      30         :                3
11025
11026 014146 000001      TABL5:  BIT0      ;BIT TO TEST CHANNEL 0 PRESENT
11027 014150 000002                      BIT1      :                1
11028 014152 000004                      BIT2      :                2
11029 014154 000010                      BIT3      :                3
11030
11031 014156 000000      ADDRESS: 0
11032 014160 000000      VECTOR: 0
11033 014162 000000      MASK: 0
11034 014164 000000      CHAN: 0
11035 014166 000000      PHASE2: 0
11036 014170 000000      P1CNT: 0          ;CTR TO PRINT 'PHASE 1' ONLY ONCE
11037
11038
  
```

```

11040
11041
11042 014172          MYTYPE:          ROUTINE MYTYPE
      (2) 014172
11043 014172          TYPTX: <<CRLF>*TEST # *>
11044 014212          TYPOCT $TESTN
11045 014220          TYPTXT <*,ERROR # *>
11046 014240 113737 001114 001176    MOV  $ITEMB,$FATAL ; APT FATAL ERROR NUMBER
11047 014246          TYPOCS $FATAL
11048 014256          TYPTXT <*,PC = *>
11049 014274          TYPOCT $ERRPC
11050 014302          TYPTXT <*,CSR: *>
11051 014320          TYPOCT DLADD
11052 014326          TYPTXT <*,VECTOR: *>
11053 014346          TYPOCT DLVEC
11054 014354 104401 001171          TYPE ,SRLF
11055 014360          ENDRTN
      (3) 014360          $255:
      (3) 014360          $256:
      (2) 014360 000207          RTS    PC
11060
  
```


11062
 11063

.SBTTL END OF PASS ROUTINE

```

(1)
(2)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1) 014362
(1) 014362 000004
(1) 014364 005037 001102
(1) 014370 005037 001160
(1) 014374 005237 001202
(1) 014400 042737 100000 001202
(1) 014406 005327
(1) 014410 000001
(1) 014412 003022
(1) 014414 012737
(1) 014416 000001
(1) 014420 014410
(1) 014422 104401 014467
(2) 014426 013746 001202
(2) 014432 104405
(1) 014434 104401 014464
(1) 014440 013700 000042
(1) 014444 001405
(1) 014446 000005
(1) 014450 004710
(1) 014452 000240
(1) 014454 000240
(1) 014456 000240
(1) 014460
(1) 014460 000137
(1) 014462 001732
(1)
(1)
(1) 014464 377 377 000
(1) 014467 015 042412 042116
(1) 014474 050040 051501 020123
(1) 014502 000043
  
```

```

*****
;*INCREMENT THE PASS NUMBER ($PASS)
;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO LOOP

$EOP:
SCOPE
CLR $TSTNM ;;ZERO THE TEST NUMBER
CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS
INC $PASS ;;INCREMENT THE PASS NUMBER
BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;;LOOP?
$EOPCT: .WORD 1
BGT $DOAGN ;;YES
MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
$ENDCT: .WORD 1
TYPE $SENDMG ;;TYPE 'END PASS #'
MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
TYPE $ENULL ;;TYPE A NULL CHARACTER
$GET42: MOV @#42,R0 ;;GET MONITOR ADDRESS
BEQ $DOAGN ;;BRANCH IF NO MONITOR
RESET ;;CLEAR THE WORLD
$ENDAD: JSR PC,(R0) ;;GO TO MONITOR
NOP ;;SAVE ROOM
NOP ;;FOR
NOP ;;ACT11
$DOAGN:
JMP @(PC)+ ;;RETURN
$RTNAD: .WORD LOOP

$ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
$SENDMG: .ASCIZ <15><12>/END PASS #/
  
```

11065
11066

.SBTTL TYPE ROUTINE

```
*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
```

```
*CALL:
*1) USING A TRAP INSTRUCTION
* TYPE ,MESADR ;:MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
```

```
*OR
* TYPE
* MESADR
*
```

```
(1) 014504 105737 001157 $TYPE: TSTB $TPFLG ;: IS THERE A TERMINAL?
(1) 014510 100002 @PL 1$ ;:BR IF YES
(1) 014512 000000 HALT ;:HALT HERE IF NO TERMINAL
(1) 014514 000430 BR 3$ ;:LEAVE
(1) 014516 010046 1$: MOV R0,-(SP) ;:SAVE R0
(1) 014520 017600 000002 MOV @2(SP),R0 ;:GET ADDRESS OF ASCIZ STRING
(1) 014524 122737 000001 001214 CMPB #APTENV,$ENV ;:RUNNING IN APT MODE
(1) 014532 001011 BNE 62$ ;:NO,GO CHECK FOR APT CONSOLE
(1) 014534 132737 000100 01215 BITB #APTPOOL,$ENVM ;:SPOOL MESSAGE TO APT
(1) 014542 001405 BEQ 62$ ;:NO,GO CHECK FOR CONSOLE
(1) 014544 010037 014554 MOV R0,61$ ;:SETUP MESSAGE ADDRESS FOR APT
(1) 014550 004737 015550 JSR PC,$ATY3 ;:SPOOL MESSAGE TO APT
(1) 014554 000000 61$: .WORD 0 ;:MESSAGE ADDRESS
(1) 014556 132737 000040 001215 62$: BITB #APTCSUP,$ENVM ;:APT CONSOLE SUPPRESSED
(1) 014564 001003 BNE 60$ ;:YES,SKIP TYPE OUT
(1) 014566 112046 2$: MOV (R0)+,-(SP) ;:PUSH CHARACTER TO BE TYPED ONTO STACK
(1) 014570 001005 BNE 4$ ;:BR IF IT ISN'T THE TERMINATOR
(1) 014572 005726 TST (SP)+ ;:IF TERMINATOR POP IT OFF THE STACK
(1) 014574 012600 60$: MOV (SP)+,R0 ;:RESTORE R0
(1) 014576 062716 000002 3$: ADD #2,(SP) ;:ADJUST RETURN PC
(1) 014602 000002 RTI ;:RETURN
(1) 014604 122716 000011 4$: CMPB #HT,(SP) ;:BRANCH IF <HT>
(1) 014610 001430 BEQ 8$ ;:BRANCH IF NOT <CRLF>
(1) 014612 122716 000200 CMPB #CRLF,(SP) ;:BRANCH IF NOT <CRLF>
(1) 014616 001006 BNE 5$ ;:POP <CR><LF> EQUIV
(1) 014620 005726 TST (SP)+ ;:TYPE A CR AND LF
(1) 014622 104401 TYPE
(1) 014624 001171 $CRLF
(1) 014626 105037 014762 CLRB $CHARCNT ;:CLEAR CHARACTER COUNT
(1) 014632 000755 BR 2$ ;:GET NEXT CHARACTER
(1) 014634 004737 014716 5$: JSR PC,$TYPEC ;:GO TYPE THIS CHARACTER
(1) 014640 123726 001156 6$: CMPB $FILLC,(SP)+ ;:IS IT TIME FOR FILLER CHARS.?
(1) 014644 001350 BNE 2$ ;:IF NO GO GET NEXT CHAR.
(1) 014646 013746 001154 MOV $NULL,-(SP) ;:GET # OF FILLER CHARS. NEEDED
;:AND THE NULL CHAR.
(1) 014652 105366 000001 7$: DECB 1(SP) ;:DOES A NULL NEED TO BE TYPED?
(1) 014656 002770 BLT 6$ ;:BR IF NO--GO POP THE NULL OFF OF STACK
(1) 014660 004737 014716 JSR PC,$TYPEC ;:GO TYPE A NULL
```

```

(1) 014664 105337 014762          DECB  $CHARCNT      ;;DO NOT COUNT AS A COUNT
(1) 014670 000770          BR      7$          ;;LOOP
(1)
(1)          ;HORIZONTAL TAB PROCESSOR
(1) 014672 112716 000040      8$:   MOVB  #' (SP)      ;;REPLACE TAB WITH SPACE
(1) 014676 004737 014716      9$:   JSR   PC,$TYPEC    ;;TYPE A SPACE
(1) 014702 132737 000007 014762  BITB  #7,$CHARCNT    ;;BRANCH IF NOT AT
(1) 014710 001372          BNE   9$          ;;TAB STOP
(1) 014712 005726          TST   (SP)+       ;;POP SPACE OFF STACK
(1) 014714 000724          BR    2$          ;;GET NEXT CHARACTER
(1) 014716 105777 164226      $TYPEC: TSTB @STPS    ;;WAIT UNTIL PRINTER IS READY
(1) 014722 100375          BPL  $TYPEC
(1) 014724 116677 000002 164220  MOVB  2(SP),@STPB  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)
(1) 014732 122766 000015 000002  CMPB  #CR,2(SP)    ;;IS CHARACTER A CARRIAGE RETURN?
(1) 014740 001003          BNE   1$          ;;BRANCH IF NO
(1) 014742 105037 014762          CLRB  $CHARCNT    ;;YES--CLEAR CHARACTER COUNT
(1) 014746 000406          BR    $TYPEX      ;;EXIT
(1) 014750 122766 000012 000002  1$:   CMPB  #LF,2(SP)  ;;IS CHARACTER A LINE FEED?
(1) 014756 001402          BEQ  $TYPEX      ;;BRANCH IF YES
(1) 014760 105227          INCB  (PC)+       ;;COUNT THE CHARACTER
(1) 014762 000000          $CHARCNT: .WORD  0 ;;CHARACTER COUNT STORAGE
(1) 014764 000207          $TYPEX: RTS      PC
(1)
(1)
(1)

```


11068
 11069

.SBTTL TTY INPUT ROUTINE

::*****

.ENABL LSB

::*****

::*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 ::*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
 ::*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
 ::*WHEN OPERATING IN TTY FLAG MODE.

| | | | | | | | |
|-----|--------|--------|--------|--------|---------------|--------------|------------------------------------|
| (1) | 014766 | 022737 | 000176 | 001140 | \$CKSWR: CMP | #SWREG,SWR | ::IS THE SOFT-SWR SELECTED? |
| (1) | 014774 | 001074 | | | BNE | 15\$ | ::BRANCH IF NO |
| (1) | 014776 | 105777 | 164142 | | TSTB | @\$TKS | ::CHAR THERE? |
| (1) | 015002 | 100071 | | | BPL | 15\$ | ::IF NO, DON'T WAIT AROUND |
| (1) | 015004 | 117746 | 164136 | | MOVB | @\$TKB,-(SP) | ::SAVE THE CHAR |
| (1) | 015010 | 042716 | 177600 | | BIC | #^C177,(SP) | ::STRIP-OFF THE ASCII |
| (1) | 015014 | 022726 | 000007 | | CMP | #7,(SP)+ | ::IS IT A CONTROL G? |
| (1) | 015020 | 001062 | | | BNE | 15\$ | ::NO, RETURN TO USER |
| (1) | 015022 | 123727 | 001134 | 000001 | CMPB | \$AUTOB,#1 | ::ARE WE RUNNING IN AUTO-MODE? |
| (1) | 015030 | 001456 | | | BEQ | 15\$ | ::BRANCH IF YES |
| (1) | 015032 | 104401 | 015513 | | TYPE | ,\$CNTLG | ::ECHO THE CONTROL-G (^G) |
| (1) | 015036 | 104401 | 015520 | | \$GTSWR: TYPE | ,\$MSWR | ::TYPE CURRENT CONTENTS |
| (2) | 015042 | 013746 | 000176 | | MOV | SWREG,-(SP) | ::SAVE SWREG FOR TYPEOUT |
| (2) | 015046 | 104402 | | | TYPOC | | ::GO TYPE--OCTAL ASCII(ALL DIGITS) |
| (1) | 015050 | 104401 | 015531 | | TYPE | ,\$MNEW | ::PROMPT FOR NEW SWR |
| (1) | 015054 | 005046 | | | 19\$: CLR | -(SP) | ::CLEAR COUNTER |
| (1) | 015056 | 005046 | | | CLR | -(SP) | |
| (1) | | | | | ::THE NEW SWR | | |
| (1) | 015060 | 105777 | 164060 | | 7\$: TSTB | @\$TKS | ::CHAR THERE? |
| (1) | 015064 | 100375 | | | BPL | 7\$ | ::IF NOT TRY AGAIN |
| (1) | | | | | | | |
| (1) | 015066 | 117746 | 164054 | | MOVB | @\$TKB,-(SP) | ::PICK UP CHAR |
| (1) | 015072 | 042716 | 177600 | | BIC | #^C177,(SP) | ::MAKE IT 7-BIT ASCII |
| (1) | | | | | | | |
| (1) | | | | | | | |
| (1) | 015076 | 021627 | 000025 | | 9\$: CMP | (SP),#25 | ::IS IT A CONTROL-U? |
| (1) | 015102 | 001005 | | | BNE | 10\$ | ::BRANCH IF NOT |
| (1) | 015104 | 104401 | 015506 | | TYPE | ,\$CNTLU | ::YES, ECHO CONTROL-U (^U) |
| (1) | 015110 | 062706 | 000006 | | 20\$: ADD | #6,SP | ::IGNORE PREVIOUS INPUT |
| (1) | 015114 | 000757 | | | BR | 19\$ | ::LET'S TRY IT AGAIN |
| (1) | | | | | | | |
| (1) | | | | | | | |
| (1) | 015116 | 021627 | 000015 | | 10\$: CMP | (SP),#15 | ::IS IT A <CR>? |
| (1) | 015122 | 001022 | | | BNE | 16\$ | ::BRANCH IF NO |
| (1) | 015124 | 005766 | 000004 | | TST | 4(SP) | ::YES, IS IT THE FIRST CHAR? |
| (1) | 015130 | 001403 | | | BEQ | 11\$ | ::BRANCH IF YES |
| (1) | 015132 | 016677 | 000002 | 164000 | MOV | 2(SP),@SWR | ::SAVE NEW SWR |
| (1) | 015140 | 062706 | 000006 | | 11\$: ADD | #6,SP | ::CLEAR UP STACK |
| (1) | 015144 | 104401 | 001171 | | 14\$: TYPE | ,\$CRLF | ::ECHO <CR> AND <LF> |
| (1) | 015150 | 123727 | 001135 | 000001 | CMPB | \$INTAG,#1 | ::RE-ENABLE TTY KBD INTERRUPTS? |
| (1) | 015156 | 001003 | | | BNE | 15\$ | ::BRANCH IF NOT |

```

(1) 015160 012777 000100 163756      MOV      #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1) 015166 000002                    RTI                    ;;RETURN
(1) 015170 004737 014716      15$:     JSR      PC,$TYPEPC  ;;ECHO CHAR
(1) 015174 021627 000060      16$:     CMP      (SP),#60    ;;CHAR < 0?
(1) 015200 002420                    BLT      18$           ;;BRANCH IF YES
(1) 015202 021627 000067      CMP      (SP),#67    ;;CHAR > 7?
(1) 015206 003015                    BGT      18$           ;;BRANCH IF YES
(1) 015210 042726 000060      BIC      #60,(SP)+   ;;STRIP-OFF ASCII
(1) 015214 005766 000002      TST      2(SP)       ;;IS THIS THE FIRST CHAR
(1) 015220 001403                    BEQ      17$           ;;BRANCH IF YES
(1) 015222 006316                    ASL      (SP)         ;;NO, SHIFT PRESENT
(1) 015224 006316                    ASL      (SP)         ;;CHAR OVER TO MAKE
(1) 015226 006316                    ASL      (SP)         ;;ROOM FOR NEW ONE.
(1) 015230 005266 000002      17$:     INC      2(SP)   ;;KEEP COUNT OF CHAR
(1) 015234 056616 177776      BIS      -2(SP),(SP) ;;SET IN NEW CHAR
(1) 015240 000707                    BR       7$           ;;GET THE NEXT ONE
(1) 015242 104401 001170      18$:     TYPE     $QUES   ;;TYPE ?<CR><LF>
(1) 015246 000720                    BR       20$          ;;SIMULATE CONTROL-U
(1)                                .DSABL  LSB
  
```

```

(1)                                ;;*****
(1)                                ;;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                ;;CALL:
(1)                                ;;*
(1)                                ;;*   RDCHR                    ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                ;;*   RETURN HERE              ;;CHARACTER IS ON THE STACK
(1)                                ;;*                                ;;WITH PARITY BIT STRIPPED OFF
(1)                                ;;*
  
```

```

(1) 015250 011646                    $RDCHR: MOV      (SP),-(SP) ;;PUSH DOWN THE PC
(1) 015252 016666 000004 000002      MOV      4(SP),2(SP)  ;;SAVE THE PS
(1) 015260 105777 163660      1$:     TSTB     @$TKS     ;;WAIT FOR
(1) 015264 100375                    BPL      1$           ;;A CHARACTER
(1) 015266 117766 163654 000004      MOVB     @$TKB,4(SP)  ;;READ THE TTY
(1) 015274 042766 177600 000004      BIC      #^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
(1) 015302 026627 000004 000023      CMP      4(SP),#23   ;;IS IT A CONTROL-S?
(1) 015310 001013                    BNE      3$           ;;BRANCH IF NO
(1) 015312 105777 163626      2$:     TSTB     @$TKS     ;;WAIT FOR A CHARACTER
(1) 015316 100375                    BPL      2$           ;;LOOP UNTIL ITS THERE
(1) 015320 117746 163622      MOVB     @$TKB,-(SP)  ;;GET CHARACTER
(1) 015324 042716 177600      BIC      #^C177,(SP) ;;MAKE IT 7-BIT ASCII
(1) 015330 022627 000021      CMP      (SP)+,#21   ;;IS IT A CONTROL-Q?
(1) 015334 001366                    BNE      2$           ;;IF NOT DISCARD IT
(1) 015336 000750                    BR       1$           ;;YES, RESUME
(1) 015340 026627 000004 000140      3$:     CMP      4(SP),#140 ;;IS IT UPPER CASE?
(1) 015346 002407                    BLT      4$           ;;BRANCH IF YES
(1) 015350 026627 000004 000175      CMP      4(SP),#175  ;;IS IT A SPECIAL CHAR?
(1) 015356 003003                    BGT      4$           ;;BRANCH IF YES
(1) 015360 042766 000040 000004      BIC      #40,4(SP)   ;;MAKE IT UPPER CASE
(1) 015366 000002      4$:     RTI                    ;;GO BACK TO USER
  
```

```

(1)                                ;;*****
(1)                                ;;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                                ;;CALL:
(1)                                ;;*
(1)                                ;;*   RDLIN                    ;;INPUT A STRING FROM THE TTY
(1)                                ;;*   RETURN HERE              ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                                ;;*                                ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)                                ;;*
  
```

```

(1)
(1) 015370 010346          SRDLIN: MOV      R3,-(SP)      ;;SAVE R3
(1) 015372 012703 015476  1$:      MOV      #$TTYIN,R3    ;;GET ADDRESS
(1) 015376 022703 015506  2$:      CMP      #$TTYIN+8.,R3  ;;BUFFER FULL?
(1) 015402 101405          BLOS     4$              ;;BR IF YES
(1) 015404 104410          RDCHR          ;;GO READ ONE CHARACTER FROM THE TTY
(1) 015406 112613          MOVB     (SP)+,(R3)      ;;GET CHARACTER
(1) 015410 122713 000177  10$:    CMPB     #177,(R3)     ;;IS IT A RUBOUT
(1) 015414 001003          BNE     3$              ;;SKIP IF NOT
(1) 015416 104401 001170  4$:      TYPE     $QUES      ;;TYPE A '?'
(1) 015422 000763          BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1) 015424 111337 015474  3$:      MOVB     (R3),9$      ;;ECHO THE CHARACTER
(1) 015430 104401 015474          TYPE     9$
(1) 015434 122723 000015          CMPB     #15,(R3)+    ;;CHECK FOR RETURN
(1) 015440 001356          BNE     2$              ;;LOOP IF NOT RETURN
(1) 015442 105063 177777          CLRB     -1(R3)       ;;CLEAR RETURN (THE 15)
(1) 015446 104401 001172          TYPE     ,SLF         ;;TYPE A LINE FEED
(1) 015452 012603          MOV      (SP)+,R3     ;;RESTORE R3
(1) 015454 011646          MOV      (SP),-(SP)   ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1) 015456 016666 000004 000002  MOV      4(SP),2(SP)  ;;FIRST ASCII CHARACTER ON IT
(1) 015464 012766 015476 000004  MOV      #$TTYIN,4(SP)
(1) 015472 000002          RTI                    ;;RETURN
(1) 015474 000          9$:      .BYTE    0            ;;STORAGE FOR ASCII CHAR. TO TYPE
(1) 015475 000          .BYTE    0            ;;TERMINATOR
(1) 015476 000010          $TTYIN: .BLKB    8.    ;;RESERVE 8 BYTES FOR TTY INPUT
(1) 015506 052536 005015 000          $CNTLU: .ASCIZ  /^U/<15><12> ;;CONTROL 'U'
(1) 015513 0136 006507 000012  $CNTLG: .ASCIZ  /^G/<15><12> ;;CONTROL 'G'
(1) 015520 005015 053523 020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
(1) 015526 020075 000
(1) 015531 040 047040 053505  $MNEW:  .ASCIZ  / NEW = /
(1) 015536 036440 000040
  
```


11071
 11072

.SBTTL APT COMMUNICATIONS ROUTINE

```

(1) (2) *****
(1) 015542 112737 000001 016006 $ATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
(1) 015550 112737 000001 016004 $ATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
(1) 015556 000403 BR $ATYC
(1) 015560 112737 000001 016006 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
(1) 015566 $ATYC:
(3) 015566 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
(3) 015570 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
(1) 015572 105737 016004 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
(1) 015576 001450 BEQ 5$ ;;IF NOT: BR
(1) 015600 122737 000001 001214 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
(1) 015606 001031 BNE 3$ ;;IF NOT: BR
(1) 015610 132737 000100 001215 BITB #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1) 015616 001425 BEQ 3$ ;;IF NOT: BR
(1) 015620 017600 000004 MOV @4(SP),R0 ;;GET MESSAGE ADDR.
(1) 015624 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
(1) 015632 005737 001174 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
(1) 015636 001375 BNE 1$ ;;IF NOT: WAIT
(1) 015640 010037 001210 MOV R0,$MSGAD
(1) ;;PUT ADDR IN MAILBOX
(1) 015644 105720 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
(1) 015646 001376 BNE 2$
(1) 015650 163700 001210 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
(1) 015654 006200 ASR R0 ;;GET MESSAGE LNGTH IN WORDS
(1) 015656 010037 001212 MOV R0,$MSGGLT ;;PUT LENGTH IN MAILBOX
(1) 015662 012737 000004 001174 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
(1) 015670 000413 BR 5$
(1) 015672 017637 000004 015716 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
(1) 015700 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
(3) 015706 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
(1) 015712 004737 014504 JSR PC,$TYPE ;;CALL TYPE MACRO
(1) 015716 000000 4$: .WORD 0
(1) 015720 5$:
(1) 015720 105737 016006 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
(1) 015724 001416 BEQ 12$ ;;IF NOT: BR
(1) 015726 005737 001214 TST $ENV ;;RUNNING UNDER APT?
(1) 015732 001413 BEQ 12$ ;;IF NOT: BR
(1) 015734 005737 001174 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
(1) 015740 001375 BNE 11$ ;;IF NOT: WAIT
(1) 015742 017637 000004 001176 MOV @4(SP),$FATAL ;;GET ERROR #
(1) 015750 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
(1) 015756 005237 001174 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
(1) 015762 105037 016006 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
(1) 015766 105037 016005 CLRB $LFLG ;;CLEAR LOG FLAG
(1) 015772 105037 016004 CLRB $MFLG ;;CLEAR MESSAGE FLAG
(3) 015776 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
(3) 016000 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
(1) 016002 000207 RTS PC ;;RETURN
(1) 016004 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
(1) 016005 000 $LFLG: .BYTE 0
(1) ;;LOG FLAG
(1) 016006 000 $FFLG: .BYTE 0 ;;FATAL FLAG
(1) 016010 .EVEN
  
```

| | | |
|-----|--------|--------------|
| (1) | 000200 | APTSIZE=200 |
| (1) | 000001 | APTENV=001 |
| (1) | 000100 | APTSPool=100 |
| (1) | 000040 | APTCSUP=040 |

11074
 11075

.SBTTL ERROR HANDLER ROUTINE

```

(1) (1) *****
(1) (1) *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
(1) (1) *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1) (1) *AND GO TO MYTYPE ON ERROR
(1) (1) *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1) (1) *SW15=1 HALT ON ERROR
(1) (1) *SW13=1 INHIBIT ERROR TYPEOUTS
(1) (1) *SW10=1 BELL ON ERROR
(1) (1) *SW09=1 LOOP ON ERROR
(1) (1) *CALL
(1) (1) * ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER

(1) (1) $ERROR:
(1) (1) 016010 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
(1) (1) 016012 105237 001103 7$: INCB SERFLG ;;SET THE ERROR FLAG
(1) (1) 016016 001775 BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
(1) (1) 016020 013777 001102 163114 MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1) (1) 016026 032777 002000 163104 BIT #BIT10,@SWR ;;BELL ON ERROR?
(1) (1) 016034 001402 BEQ 1$ ;;NO - SKIP
(1) (1) 016036 104401 001164 TYPE ,SBELL ;;RING BELL
(1) (1) 016042 005237 001112 1$: INC $ERTTL ;;COUNT THE NUMBER OF ERRORS
(1) (1) 016046 011637 001116 MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
(1) (1) 016052 162737 000002 001116 SUB #2,$ERRPC
(1) (1) 016060 117737 163032 001114 MOVB @ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
(1) (1) 016066 032777 020000 163044 BIT #BIT13,@SWR ;;SKIP TYPEOUT IF SET
(1) (1) 016074 001004 BNE 20$ ;;SKIP TYPEOUTS
(1) (1) 016076 004737 014172 JSR PC,MYTYPE ;;GO TO USER ERROR ROUTINE
(1) (1) 016102 104401 001171 TYPE ,SCRLF

(1) (1) 016106 122737 000001 001214 20$: CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
(1) (1) 016114 001007 BNE 2$ ;;NO,SKIP APT ERROR REPORT
(1) (1) 016116 113737 001114 016130 MOVB $ITEMB,21$ ;;SET ITEM NUMBER AS ERROR NUMBER
(1) (1) 016124 004737 015560 JSR PC,$ATY4 ;;REPORT FATAL ERROR TO APT

(1) (1) 016130 000 .BYTE 0 21$: .BYTE 0
(1) (1) 016131 000 .BYTE 0
(1) (1) 016132 000777 22$: BR 22$ ;;APT ERROR LOOP
(1) (1) 016134 005777 163000 2$: TST @SWR ;;HALT ON ERROR
(1) (1) 016140 100002 BPL 3$ ;;SKIP IF CONTINUE
(1) (1) 016142 000000 HALT ;;HALT ON ERROR!
(1) (1) 016144 104407 CKSWR ;;TEST FOR CHANGE IN SOFT-SWR
(1) (1) 016146 032777 001000 162764 3$: BIT #BIT09,@SWR ;;LOOP ON ERROR SWITCH SET?
(1) (1) 016154 001402 BEQ 4$ ;;BR IF NO
(1) (1) 016156 013716 001110 MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
(1) (1) 016162 005737 001162 4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
(1) (1) 016166 001402 BEQ 5$ ;;BR IF NONE
(1) (1) 016170 013716 001162 MOV $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1) (1) 016174 022737 014450 000042 5$: CMP #SENDAD,@#42 ;;ACT-11 AUTO-ACCEPT?
(1) (1) 016202 001001 BNE 6$ ;;BRANCH IF NO
(1) (1) 016204 000000 HALT ;;YES
(1) (1) 016206 000002 6$: RTI ;;RETURN
  
```


| | | | | | | | |
|-----|--------|--------|--------|--------|----------------|-------------------|--|
| (1) | 016434 | 011637 | 001110 | | MOV | (SP), \$LPERR | ::SAVE ERROR LOOP ADDRESS |
| (1) | 016440 | 005037 | 001162 | | CLR | \$ESCAPE | ::CLEAR THE ESCAPE FROM ERROR ADDRESS |
| (1) | 016444 | 112737 | 000001 | 001115 | MOVB | #1, \$ERMAX | ::ONLY ALLOW ONE(1) ERROR ON NEXT TEST |
| (1) | 016452 | 013777 | 001102 | 162462 | \$OVER: MOV | \$TSTNM, @DISPLAY | ::DISPLAY TEST NUMBER |
| (1) | 016460 | 013716 | 001106 | | MOV | \$LPADR, (SP) | ::FUDGE RETURN ADDRESS |
| (1) | 016464 | 000002 | | | RTI | | ::FIXES PS |
| (1) | 016466 | 003720 | | | \$MXCNT: 2000. | | ::MAX. NUMBER OF ITERATIONS |

11080
11081

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

(1) (1) *****
(2) (1) : *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
(1) (1) : *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
(1) (1) : *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
(1) (1) : *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
(1) (1) : *REPLACED WITH SPACES.
(1) (1) : *CALL:
(1) (1) : *      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
(1) (1) : *      TYPDS                      ;;GO TO THE ROUTINE

(1) (1) $TYPDS:
(3) (3) 016470 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
(3) (3) 016472 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
(3) (3) 016474 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
(3) (3) 016476 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
(3) (3) 016500 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
(1) (1) 016502 012746      MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
(1) (1) 016506 016605      MOV      20(SP),R5     ;;GET THE INPUT NUMBER
(1) (1) 016512 100004      BPL      1$           ;;BR IF INPUT IS POS.
(1) (1) 016514 005405      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
(1) (1) 016516 112766      MOVB     #'-,1(SP)    ;;MAKE THE ASCII NUMBER NEG.
(1) (1) 016524 005000      CLR      R0           ;;ZERO THE CONSTANTS INDEX
(1) (1) 016526 012703      MOV      #$DBLK,R3    ;;SETUP THE OUTPUT POINTER
(1) (1) 016532 112723      MOVB     #' ,(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK
(1) (1) 016536 005002      CLR      R2           ;;CLEAR THE BCD NUMBER
(1) (1) 016540 016001      MOV      $DTBL(R0),R1 ;;GET THE CONSTANT
(1) (1) 016544 160105      SUB      R1,R5        ;;FORM THIS BCD DIGIT
(1) (1) 016546 002402      BLT      4$           ;;BR IF DONE
(1) (1) 016550 005202      INC      R2           ;;INCREASE THE BCD DIGIT BY 1
(1) (1) 016552 000774      BR       3$
(1) (1) 016554 060105      ADD      R1,R5        ;;ADD BACK THE CONSTANT
(1) (1) 016556 005702      TST      R2           ;;CHECK IF BCD DIGIT=0
(1) (1) 016560 001002      BNE      5$           ;;FALL THROUGH IF 0
(1) (1) 016562 105716      TSTB     (SP)         ;;STILL DOING LEADING 0'S?
(1) (1) 016564 100407      BMI      7$           ;;BR IF YES
(1) (1) 016566 106316      ASLB     (SP)         ;;MSD?
(1) (1) 016570 103003      BCC      6$           ;;BR IF NO
(1) (1) 016572 116663      MOVB     1(SP),-1(R3)  ;;YES--SET THE SIGN
(1) (1) 016600 052702      BIS      #'0,R2       ;;MAKE THE BCD DIGIT ASCII
(1) (1) 016604 052702      BIS      #' ,R2       ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1) (1) 016610 110223      MOVB     R2,(R3)+     ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1) (1) 016612 005720      TST      (R0)+        ;;JUST INCREMENTING
(1) (1) 016614 020027      CMP      R0,#10       ;;CHECK THE TABLE INDEX
(1) (1) 016620 002746      BLT      2$           ;;GO DO THE NEXT DIGIT
(1) (1) 016622 003002      BGT      8$           ;;GO TO EXIT
(1) (1) 016624 010502      MOV      R5,R2        ;;GET THE LSD
(1) (1) 016626 000764      BR       6$           ;;GO CHANGE TO ASCII
(1) (1) 016630 105726      TSTB     (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
(1) (1) 016632 100003      BPL      9$           ;;BR IF NO
(1) (1) 016634 116663      MOVB     -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
(1) (1) 016642 105013      CLR      (R3)         ;;SET THE TERMINATOR
(3) (3) 016644 012605      MOV      (SP)+,R5     ;;POP STACK INTO R5
(3) (3) 016646 012603      MOV      (SP)+,R3     ;;POP STACK INTO R3

```


| | | | | | | | |
|-----|--------|--------|--------|--------|---------|-------------|-----------------------|
| (3) | 016650 | 012602 | | | MOV | (SP)+,R2 | ::POP STACK INTO R2 |
| (3) | 016652 | 012601 | | | MOV | (SP)+,R1 | ::POP STACK INTO R1 |
| (3) | 016654 | 012600 | | | MOV | (SP)+,R0 | ::POP STACK INTO R0 |
| (1) | 016656 | 104401 | 016704 | | TYPE | ,SDBLK | ::NOW TYPE THE NUMBER |
| (1) | 016662 | 016666 | 000002 | 000004 | MOV | 2(SP),4(SP) | ::ADJUST THE STACK |
| (1) | 016670 | 012616 | | | MOV | (SP)+,(SP) | |
| (1) | 016672 | 000002 | | | RTI | | ::RETURN TO USER |
| (1) | 016674 | 023420 | | | \$DTBL: | 10000. | |
| (1) | 016676 | 001750 | | | | 1000. | |
| (1) | 016700 | 000144 | | | | 100. | |
| (1) | 016702 | 000012 | | | | 10. | |
| (1) | 016704 | 000004 | | | \$DBLK: | .BLKW 4 | |

11083
11084

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

(1) (1) *****
(1) (1) *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1) (1) *OCTAL (ASCII) NUMBER AND TYPE IT.
(1) (1) *$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1) (1) *CALL:
(1) (1) *       MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
(1) (1) *       TYPOS    ;;CALL FOR TYPEOUT
(1) (1) *       .BYTE   N        ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1) (1) *       .BYTE   M        ;;M=1 OR 0
(1) (1) *                               ;;1=TYPE LEADING ZEROS
(1) (1) *                               ;;0=SUPPRESS LEADING ZEROS
(1) (1) *$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1) (1) *$TYPOS OR $TYPOC
(1) (1) *CALL:
(1) (1) *       MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
(1) (1) *       TYPON    ;;CALL FOR TYPEOUT
(1) (1) *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1) (1) *CALL:
(1) (1) *       MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
(1) (1) *       TYPOC    ;;CALL FOR TYPEOUT
(1) (1) 016714 017646 000000   $TYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
(1) (1) 016720 116637 000001   017137  MOV      1(SP), $OFILL    ;;LOAD ZERO FILL SWITCH
(1) (1) 016726 112637 017141   MOV      (SP)+, $OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1) (1) 016732 062716 000002   ADD      #2,(SP)        ;;ADJUST RETURN ADDRESS
(1) (1) 016736 000406                BR      $TYPON
(1) (1) 016740 112737 000001   017137 $TYPOC: MOV      #1, $OFILL    ;;SET THE ZERO FILL SWITCH
(1) (1) 016746 112737 000006   017141 MOV      #6, $OMODE+1    ;;SET FOR SIX(6) DIGITS
(1) (1) 016754 112737 000005   017136 $TYPON: MOV      #5, $OCNT    ;;SET THE ITERATION COUNT
(1) (1) 016762 010346                MOV      R3, -(SP)      ;;SAVE R3
(1) (1) 016764 010446                MOV      R4, -(SP)      ;;SAVE R4
(1) (1) 016766 010546                MOV      R5, -(SP)      ;;SAVE R5
(1) (1) 016770 113704 017141   MOV      $OMODE+1, R4    ;;GET THE NUMBER OF DIGITS TO TYPE
(1) (1) 016774 005404                NEG      R4
(1) (1) 016776 062704 000006   ADD      #6, R4         ;;SUBTRACT IT FOR MAX. ALLOWED
(1) (1) 017002 110437 017140   MOV      R4, $OMODE      ;;SAVE IT FOR USE
(1) (1) 017006 113704 017137   MOV      $OFILL, R4     ;;GET THE ZERO FILL SWITCH
(1) (1) 017012 016605 000012   MOV      12(SP), R5     ;;PICKUP THE INPUT NUMBER
(1) (1) 017016 005003                CLR      R3             ;;CLEAR THE OUTPUT WORD
(1) (1) 017020 006105                $1:   ROL      R5         ;;ROTATE MSB INTO 'C'
(1) (1) 017022 000404                BR      3$             ;;GO DO MSB
(1) (1) 017024 006105                $2:   ROL      R5         ;;FORM THIS DIGIT
(1) (1) 017026 006105                ROL      R5
(1) (1) 017030 006105                ROL      R5
(1) (1) 017032 010503                MOV      R5, R3
(1) (1) 017034 006103                $3:   ROL      R3         ;;GET LSB OF THIS DIGIT
(1) (1) 017036 105337 017140   DEC      $OMODE         ;;TYPE THIS DIGIT?
(1) (1) 017042 100016                BPL      7$             ;;BR IF NO
(1) (1) 017044 042703 177770   BIC      #177770, R3    ;;GET RID OF JUNK
(1) (1) 017050 001002                BNE      4$             ;;TEST FOR 0
(1) (1) 017052 005704                TST     R4             ;;SUPPRESS THIS 0?

```

| | | | | | | | |
|-----|--------|--------|--------|--------|----------|-------------|-----------------------------------|
| (1) | 017054 | 001403 | | | BEO | 5\$ | ::BR IF YES |
| (1) | 017056 | 005204 | | | INC | R4 | ::DON'T SUPPRESS ANYMORE 0'S |
| (1) | 017060 | 052703 | 000060 | | BIS | #'0,R3 | ::MAKE THIS DIGIT ASCII |
| (1) | 017064 | 052703 | 000040 | | BIS | #' R3 | ::MAKE ASCII IF NOT ALREADY |
| (1) | 017070 | 110337 | 017134 | | MOVB | R3,8\$ | ::SAVE FOR TYPING |
| (1) | 017074 | 104401 | 017134 | | TYPE | .8\$ | ::GO TYPE THIS DIGIT |
| (1) | 017100 | 105337 | 017136 | | DECB | \$OCNT | ::COUNT BY 1 |
| (1) | 017104 | 003347 | | | BGT | 2\$ | ::BR IF MORE TO DO |
| (1) | 017106 | 002402 | | | BLT | 6\$ | ::BR IF DONE |
| (1) | 017110 | 005204 | | | INC | R4 | ::INSURE LAST DIGIT ISN'T A BLANK |
| (1) | 017112 | 000744 | | | BR | 2\$ | ::GO DO THE LAST DIGIT |
| (1) | 017114 | 012605 | | | MOV | (SP)+,R5 | ::RESTORE R5 |
| (1) | 017116 | 012604 | | | MOV | (SP)+,R4 | ::RESTORE R4 |
| (1) | 017120 | 012603 | | | MOV | (SP)+,R3 | ::RESTORE R3 |
| (1) | 017122 | 016666 | 000002 | 000004 | MOV | 2(SP),4(SP) | ::SET THE STACK FOR RETURNING |
| (1) | 017130 | 012616 | | | MOV | (SP)+,(SP) | |
| (1) | 017132 | 000002 | | | RTI | | ::RETURN |
| (1) | 017134 | 000 | | | .BYTE | 0 | ::STORAGE FOR ASCII DIGIT |
| (1) | 017135 | 000 | | | .BYTE | 0 | ::TERMINATOR FOR TYPE ROUTINE |
| (1) | 017136 | 000 | | | \$OCNT: | .BYTE 0 | ::OCTAL DIGIT COUNTER |
| (1) | 017137 | 000 | | | \$OFILL: | .BYTE 0 | ::ZERO FILL SWITCH |
| (1) | 017140 | 000000 | | | \$OMODE: | .WORD 0 | ::NUMBER OF DIGITS TO TYPE |

11086
11087

.SBTTL TRAP DECODER

(1)
(2)
(1)
(1)
(1)
(1)
(1)
(1) 017142 010046
(1) 017144 016600 000002
(1) 017150 005740
(1) 017152 111000
(1) 017154 006300
(1) 017156 016000 017176
(1) 017162 000200
(1)
(1)
(1)
(1)
(1) 017164 011646
(1) 017166 016666 000004 000002
(1) 017174 000002
(1)
(3)
(3)
(3)
(3)
(3)
(3)
(3) 017176 017164
(3) 017200 014504
(3) 017202 016740
(3) 017204 016714
(3) 017206 016754
(3) 017210 016470
(1)
(3) 017212 015036
(1)
(3) 017214 014766
(3) 017216 015250
(3) 017220 015370

: THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
: AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
: OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
: GO TO THAT ROUTINE.

\$TRAP: MOV R0,-(SP) ;;SAVE R0
MOV 2(SP),R0 ;;GET TRAP ADDRESS
TST -(R0) ;;BACKUP BY 2
MOVB (R0),R0 ;;GET RIGHT BYTE OF TRAP
ASL R0 ;;POSITION FOR INDEXING
MOV \$TRPAD(R0),R0 ;;INDEX TO TABLE
RTS R0 ;;GO TO ROUTINE

;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

\$TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
RTI ;;RESTORE THE PSW

.SBTTL TRAP TABLE

: THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
: BY THE "TRAP" INSTRUCTION.

: ROUTINE
:-----
\$TRPAD: .WORD \$TRAP2
\$TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
\$TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
\$TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
\$TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
\$TYPDS ;;CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
\$GTSWR ;;CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
\$CKSWR ;;CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
\$RDCHR ;;CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
\$RDLIN ;;CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE

:MAXMEM FOR APT = 17400
:MAXMEM NON APT = 17500 (ABS LOADER)
:THE FOLLOWING CALL TO CNMAC2.SML WAS ADDED TO INIT SBC 11/21

11088
11089
11090
11091
11092
11093

(1) 017222
(1) 000100 000100
(1) 000102 017222 000300
(1) 000140 000140
(1) 000142 170000 000300
(1) 017222

:SPECIFICS
POINT=.; SAVE POINTER
.=100
\$CLKVEC ;LKVEC HANDLER
300 ;INTERRUPT HANDLER PRI
.=140 ;BRKVEC
170000 ;ODT START ADDRESS
300 ;PRIORITY
.=POINT ;RESTORE POINTER

| | | | | | |
|-------|--------|--------|--------|-----------|--|
| (1) | 017222 | 104401 | 017230 | \$CLKVEC: | TYPE,CLKMES |
| (1) | 017226 | 000000 | | | HALT |
| (1) | 017230 | 005015 | 045514 | 042526 | CLKMES: .ASCIZ <15><12>/LKVEC INTERRUPT - DISCONNECT LTC / |
| (1) | 017236 | 020103 | 047111 | 042524 | |
| (1) | 017244 | 051122 | 050125 | 020124 | |
| (1) | 017252 | 020055 | 044504 | 041523 | |
| (1) | 017260 | 047117 | 042516 | 052103 | |
| (1) | 017266 | 046040 | 041524 | 000040 | |
| 11094 | | 000001 | | | .END |

| | | | | | | | | | | | | | | | | | | | |
|---------|----------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--|--|--|--|--|
| SW01 | = 000002 | 8724# | | | | | | | | | | | | | | | | | |
| SW02 | = 000004 | 8724# | | | | | | | | | | | | | | | | | |
| SW03 | = 000010 | 8724# | | | | | | | | | | | | | | | | | |
| SW04 | = 000020 | 8724# | | | | | | | | | | | | | | | | | |
| SW05 | = 000040 | 8724# | | | | | | | | | | | | | | | | | |
| SW06 | = 000100 | 8724# | | | | | | | | | | | | | | | | | |
| SW07 | = 000200 | 8724# | | | | | | | | | | | | | | | | | |
| SW08 | = 000400 | 8724# | | | | | | | | | | | | | | | | | |
| SW09 | = 001000 | 8724# | | | | | | | | | | | | | | | | | |
| SW1 | = 000002 | 8724# | | | | | | | | | | | | | | | | | |
| SW10 | = 002000 | 8724# | | | | | | | | | | | | | | | | | |
| SW11 | = 004000 | 8724# | | | | | | | | | | | | | | | | | |
| SW12 | = 010000 | 8724# | | | | | | | | | | | | | | | | | |
| SW13 | = 020000 | 8724# | | | | | | | | | | | | | | | | | |
| SW14 | = 040000 | 8724# | | | | | | | | | | | | | | | | | |
| SW15 | = 100000 | 8724# | | | | | | | | | | | | | | | | | |
| SW2 | = 000004 | 8724# | | | | | | | | | | | | | | | | | |
| SW3 | = 000010 | 8724# | | | | | | | | | | | | | | | | | |
| SW4 | = 000020 | 8724# | | | | | | | | | | | | | | | | | |
| SW5 | = 000040 | 8724# | | | | | | | | | | | | | | | | | |
| SW6 | = 000100 | 8724# | | | | | | | | | | | | | | | | | |
| SW7 | = 000200 | 8724# | | | | | | | | | | | | | | | | | |
| SW8 | = 000400 | 8724# | | | | | | | | | | | | | | | | | |
| SW9 | = 001000 | 8724# | | | | | | | | | | | | | | | | | |
| TABEND | 007764 | 10081 | 10312# | | | | | | | | | | | | | | | | |
| TABL1 | 013520 | 10817 | 10886# | | | | | | | | | | | | | | | | |
| TABL2 | 014116 | 10968 | 10969 | 11011# | | | | | | | | | | | | | | | |
| TABL3 | 014126 | 10970 | 11016# | | | | | | | | | | | | | | | | |
| TABL4 | 014136 | 10995 | 10996 | 11021# | | | | | | | | | | | | | | | |
| TABL5 | 014146 | 10997 | 11026# | | | | | | | | | | | | | | | | |
| TABL6 | 012416 | 10457 | 10655# | | | | | | | | | | | | | | | | |
| TABL7 | 012426 | 10416 | 10417 | 10660# | | | | | | | | | | | | | | | |
| TABL8 | 012436 | 10419 | 10420 | 10665# | | | | | | | | | | | | | | | |
| TBITVE= | 000014 | 8724# | | | | | | | | | | | | | | | | | |
| TBUF | 001266 | 8864# | 8899* | 8964 | 9207* | 9219* | 9227* | 9288* | 9333* | 9376* | 9430* | 9441* | 9489* | 9581* | | | | | |
| | | 9623* | 9662* | 9708* | 9856* | 9932* | 10006* | | | | | | | | | | | | |
| TCH0 | 007734 | 10219 | 10301# | | | | | | | | | | | | | | | | |
| TCH1 | 007736 | 10239 | 10302# | | | | | | | | | | | | | | | | |
| TCH2 | 007740 | 10259 | 10303# | | | | | | | | | | | | | | | | |
| TCH3 | 007742 | 10279 | 10304# | | | | | | | | | | | | | | | | |
| TCNT | 007772 | 10152* | 10162* | 10169* | 10170* | 10172* | 10174 | 10317# | | | | | | | | | | | |
| TCSR | 001264 | 8863# | 8898* | 8960 | 8999 | 9007* | 9009 | 9018* | 9020 | 9029* | 9032 | 9055 | 9063* | 9065 | | | | | |
| | | 9074* | 9076 | 9085* | 9088 | 9165 | 9211 | 9220 | 9231 | 9241 | 9545 | 9548* | 9554* | 9576* | | | | | |
| | | 9627 | 9700 | 9801* | 9812* | 9859* | 9890* | 9930* | 9988* | 9990 | 9999* | | | | | | | | |
| TDATA0= | 000001 | 8820# | | | | | | | | | | | | | | | | | |
| TDATA1= | 000002 | 8819# | | | | | | | | | | | | | | | | | |
| TDATA2= | 000004 | 8818# | | | | | | | | | | | | | | | | | |
| TDATA3= | 000010 | 8817# | | | | | | | | | | | | | | | | | |
| TDATA4= | 000020 | 8816# | | | | | | | | | | | | | | | | | |
| TDATA5= | 000040 | 8815# | | | | | | | | | | | | | | | | | |
| TDATA6= | 000100 | 8814# | | | | | | | | | | | | | | | | | |
| TDATA7= | 000200 | 8813# | | | | | | | | | | | | | | | | | |
| TEMP | 007774 | 10210* | 10211* | 10213* | 10221* | 10222* | 10224* | 10231* | 10232* | 10234* | 10241* | 10242* | 10244* | 10251* | | | | | |
| | | 10252* | 10254* | 10261* | 10262* | 10264* | 10271* | 10272* | 10274* | 10281* | 10282* | 10284* | 10318# | | | | | | |
| TIME = | 000000 | 10768# | 10770 | | | | | | | | | | | | | | | | |
| TIMER | 012446 | 9211 | 9241 | 9293 | 9337 | 9380 | 9432 | 9490 | 9545 | 9556 | 9583 | 9627 | 9632 | 9665 | | | | | |

| | | | | | | | | | | | | | | | | | | | | |
|----------|---------|---------|--------|--------|-------|--------|-------|-------|-------|-------|-------|--------|--------|--------|--|--|--|--|--|--|
| XMITRD= | 000200 | 8792# | 9165 | 9211 | 9231 | 9241 | 9545 | 9700 | | | | | | | | | | | | |
| XMITOS | 007454 | 10217# | | | | | | | | | | | | | | | | | | |
| XMIT1S | 007534 | 10237# | | | | | | | | | | | | | | | | | | |
| XMIT2S | 007614 | 10257# | | | | | | | | | | | | | | | | | | |
| XMIT3S | 007674 | 10277# | | | | | | | | | | | | | | | | | | |
| XOSRV | 010734 | 10533# | | | | | | | | | | | | | | | | | | |
| X1SRV | 011054 | 10560# | | | | | | | | | | | | | | | | | | |
| X2SRV | 011174 | 10587# | | | | | | | | | | | | | | | | | | |
| X3SRV | 011314 | 10614# | | | | | | | | | | | | | | | | | | |
| \$APTHD | 001000 | 8856# | | | | | | | | | | | | | | | | | | |
| \$ASTAT= | ***** U | 11072 | | | | | | | | | | | | | | | | | | |
| \$ATYC | 015566 | 11072# | | | | | | | | | | | | | | | | | | |
| \$ATY1 | 015542 | 11072# | | | | | | | | | | | | | | | | | | |
| \$ATY3 | 015550 | 11066 | 11072# | | | | | | | | | | | | | | | | | |
| \$ATY4 | 015560 | 11072# | 11075 | | | | | | | | | | | | | | | | | |
| \$AUTOB | 001134 | 8857# | 8872* | 11069 | | | | | | | | | | | | | | | | |
| \$BASE | 001250 | 8857# | 10789 | 10955 | | | | | | | | | | | | | | | | |
| \$BDADR | 001122 | 8857# | | | | | | | | | | | | | | | | | | |
| \$BDDAT | 001126 | 8857# | | | | | | | | | | | | | | | | | | |
| \$BELL | 001164 | 8857# | 11075 | | | | | | | | | | | | | | | | | |
| \$BGNLE= | 177777 | 8612# | | | | | | | | | | | | | | | | | | |
| \$CHARC | 014762 | 11066#* | | | | | | | | | | | | | | | | | | |
| \$CKSWR | 014766 | 11069# | 11087 | | | | | | | | | | | | | | | | | |
| \$CLKVE | 017222 | 11093# | | | | | | | | | | | | | | | | | | |
| \$CMTAG | 001100 | 8857# | 8871 | | | | | | | | | | | | | | | | | |
| \$CM3 = | 000000 | 8857# | | | | | | | | | | | | | | | | | | |
| \$CNTLG | 015513 | 11069# | | | | | | | | | | | | | | | | | | |
| \$CNTLU | 015506 | 11069# | | | | | | | | | | | | | | | | | | |
| \$CPUOP | 001222 | 8857# | | | | | | | | | | | | | | | | | | |
| \$CRLF | 001171 | 8857# | 10049 | 10873 | 11054 | 11066 | 11069 | 11075 | | | | | | | | | | | | |
| \$DBLK | 016704 | 11081# | | | | | | | | | | | | | | | | | | |
| \$DEVCT | 001204 | 8857# | 8905* | | | | | | | | | | | | | | | | | |
| \$DEVN | 001252 | 8857# | 10783 | 10796* | 10805 | 10817* | 10824 | 10825 | 10826 | 10827 | | | | | | | | | | |
| \$DOAGN | 014460 | 11063# | | | | | | | | | | | | | | | | | | |
| \$DTBL | 016674 | 11081# | | | | | | | | | | | | | | | | | | |
| \$ENDAD | 014450 | 8854 | 8872 | 11063# | 11075 | | | | | | | | | | | | | | | |
| \$ENDCT | 014416 | 8871 | 11063# | | | | | | | | | | | | | | | | | |
| \$ENDMG | 014467 | 11063# | | | | | | | | | | | | | | | | | | |
| \$ENULL | 014464 | 11063# | | | | | | | | | | | | | | | | | | |
| \$ENV | 001214 | 8857# | 8872 | 11066 | 11072 | 11075 | | | | | | | | | | | | | | |
| \$ENVN | 001215 | 8857# | 8871 | 11066 | 11072 | | | | | | | | | | | | | | | |
| \$EOP | 014362 | 10822 | 10964 | 11063# | | | | | | | | | | | | | | | | |
| \$EOPCT | 014410 | 8871* | 11063# | | | | | | | | | | | | | | | | | |
| \$ERFLG | 001103 | 8857# | 11075* | 11078* | | | | | | | | | | | | | | | | |
| \$ERMAX | 001115 | 8857# | 8871* | 11078* | | | | | | | | | | | | | | | | |
| \$ERRFL= | 000000 | 8892# | 8894# | 8896# | 8897# | 8898# | 8899# | 8900# | 8905# | 8909# | 8928# | 8933# | 8935# | 8937# | | | | | | |
| | | 8946# | 8947# | 8997# | 9006# | 9007# | 9016# | 9018# | 9027# | 9029# | 9053# | 9062# | 9063# | 9072# | | | | | | |
| | | 9074# | 9083# | 9085# | 9110# | 9119# | 9120# | 9129# | 9131# | 9140# | 9142# | 9163# | 9196# | 9199# | | | | | | |
| | | 9200# | 9207# | 9211# | 9227# | 9234# | 9241# | 9248# | 9253# | 9256# | 9285# | 9288# | 9293# | 9303# | | | | | | |
| | | 9329# | 9333# | 9337# | 9349# | 9372# | 9376# | 9380# | 9393# | 9425# | 9430# | 9432# | 9441# | 9443# | | | | | | |
| | | 9446# | 9459# | 9471# | 9483# | 9489# | 9490# | 9534# | 9537# | 9539# | 9543# | 9545# | 9548# | 9554# | | | | | | |
| | | 9556# | 9565# | 9574# | 9576# | 9578# | 9581# | 9583# | 9612# | 9614# | 9615# | 9617# | 9623# | 9627# | | | | | | |
| | | 9629# | 9632# | 9640# | 9655# | 9658# | 9660# | 9662# | 9665# | 9695# | 9700# | 9708# | 9710# | 9718# | | | | | | |
| | | 9780# | 9782# | 9783# | 9786# | 9787# | 9790# | 9791# | 9793# | 9795# | 9801# | 9803# | 9812# | 9813# | | | | | | |
| | | 9856# | 9859# | 9877# | 9881# | 9882# | 9883# | 9885# | 9887# | 9890# | 9926# | 9928# | 9930# | 9932# | | | | | | |
| | | 9934# | 9939# | 9942# | 9945# | 9953# | 9957# | 9986# | 9988# | 9990# | 9999# | 10000# | 10004# | 10006# | | | | | | |

| | | | | | | | | | | | | |
|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 10008# | 10040# | 10053# | 10054# | 10078# | 10080# | 10087# | 10088# | 10090# | 10091# | 10092# | 10099# | 10100# |
| 10101# | 10105# | 10106# | 10108# | 10110# | 10112# | 10114# | 10122# | 10124# | 10125# | 10127# | 10129# | 10144# |
| 10152# | 10154# | 10155# | 10157# | 10208# | 10210# | 10211# | 10213# | 10219# | 10221# | 10222# | 10224# | 10229# |
| 10231# | 10232# | 10234# | 10239# | 10241# | 10242# | 10244# | 10249# | 10251# | 10252# | 10254# | 10259# | 10261# |
| 10262# | 10264# | 10269# | 10271# | 10272# | 10274# | 10279# | 10281# | 10282# | 10284# | 10344# | 10348# | 10350# |
| 10354# | 10355# | 10357# | 10358# | 10359# | 10360# | 10361# | 10362# | 10369# | 10371# | 10372# | 10374# | 10378# |
| 10380# | 10381# | 10383# | 10384# | 10386# | 10388# | 10391# | 10393# | 10395# | 10401# | 10402# | 10403# | 10404# |
| 10409# | 10411# | 10413# | 10416# | 10417# | 10419# | 10420# | 10422# | 10426# | 10434# | 10436# | 10444# | 10445# |
| 10446# | 10447# | 10449# | 10450# | 10453# | 10455# | 10457# | 10459# | 10461# | 10462# | 10464# | 10468# | 10472# |
| 10475# | 10477# | 10481# | 10482# | 10484# | 10487# | 10519# | 10520# | 10521# | 10524# | 10526# | 10529# | 10534# |
| 10535# | 10536# | 10537# | 10539# | 10542# | 10546# | 10547# | 10548# | 10551# | 10553# | 10556# | 10561# | 10562# |
| 10563# | 10564# | 10566# | 10569# | 10573# | 10574# | 10575# | 10578# | 10580# | 10583# | 10588# | 10589# | 10590# |
| 10591# | 10593# | 10596# | 10600# | 10601# | 10602# | 10605# | 10607# | 10610# | 10615# | 10616# | 10617# | 10618# |
| 10620# | 10623# | 10705# | 10706# | 10707# | 10715# | 10717# | 10723# | 10730# | 10731# | 10739# | 10741# | 10770# |
| 10771# | 10772# | 10931# | | | | | | | | | | |

\$ERROR 016010
 \$ERRPC 001116
 \$ERRTB 001254
 \$ERTTL 001112
 \$ESCAP 001162
 \$ETABL 001214
 \$ETEND 001254
 \$FATAL 001176
 \$FFLG 016006
 \$FILLC 001156
 \$FILLS 001155
 \$FSAND= 000310

| | | | | | | | | | | | | |
|---------|--------|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 8871 | 11075# | | | | | | | | | | | |
| 8857# | 11049 | 11075* | | | | | | | | | | |
| 8857# | | | | | | | | | | | | |
| 8857# | 10048 | 10051* | 11075* | | | | | | | | | |
| 8857# | 8871* | 11075 | 11078* | | | | | | | | | |
| 8857# | | | | | | | | | | | | |
| 8856 | 8857# | | | | | | | | | | | |
| 8857# | 11046* | 11047 | 11072* | | | | | | | | | |
| 11072#* | | | | | | | | | | | | |
| 8857# | 11066 | | | | | | | | | | | |
| 8857# | 11066 | | | | | | | | | | | |
| 8612# | 8906 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 |
| 9122 | 9133 | 9145 | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 |
| 9395 | 9413 | 9414 | 9449 | 9460 | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 |
| 9727 | 9728 | 9729 | 9739 | 9758 | 9792 | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 |
| 9886 | 9918 | 9921 | 9940 | 9944 | 9947 | 9949 | 9951 | 9956 | 9968 | 9971 | 9974 | 9979 |
| 10015 | 10034 | 10035 | 10038 | 10052 | 10103 | 10126 | 10128 | 10181 | 10376 | 10392 | 10394 | 10415 |
| 10452 | 10467 | 10471 | 10483 | 10485 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 | 10714 | 10721 |
| 10738 | | | | | | | | | | | | |

\$FSBAD= 000401
 \$FSBLA= 000170
 \$FSCAS= 000150
 \$F\$DEC= 000220
 \$F\$G00= 000400

| | | | | | | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 8612# | 8906 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 |
| 9122 | 9133 | 9145 | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 |
| 9395 | 9413 | 9414 | 9449 | 9460 | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 |
| 9727 | 9728 | 9729 | 9739 | 9758 | 9792 | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 |
| 9886 | 9918 | 9921 | 9940 | 9944 | 9947 | 9949 | 9951 | 9956 | 9968 | 9971 | 9974 | 9979 |
| 10015 | 10034 | 10035 | 10038 | 10052 | 10103 | 10126 | 10128 | 10181 | 10376 | 10392 | 10394 | 10415 |
| 10452 | 10467 | 10471 | 10483 | 10485 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 | 10714 | 10721 |
| 10738 | | | | | | | | | | | | |
| 8612# | | | | | | | | | | | | |
| 8612# | | | | | | | | | | | | |
| 8612# | | | | | | | | | | | | |
| 8612# | 8906 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 |
| 9122 | 9133 | 9145 | 9165 | 9192 | 9214 | 9231 | 9242 | 9247 | 9249 | 9281 | 9296 | 9307 |
| 9325 | 9339 | 9350 | 9368 | 9382 | 9395 | 9413 | 9414 | 9435 | 9449 | 9460 | 9474 | 9493 |
| 9498 | 9507 | 9559 | 9568 | 9606 | 9635 | 9643 | 9667 | 9689 | 9701 | 9711 | 9720 | 9727 |
| 9728 | 9729 | 9739 | 9758 | 9792 | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 | 9886 |
| 9918 | 9921 | 9935 | 9940 | 9944 | 9947 | 9949 | 9951 | 9956 | 9968 | 9971 | 9974 | 9979 |
| 9993 | 10009 | 10015 | 10034 | 10035 | 10038 | 10052 | 10103 | 10116 | 10126 | 10128 | 10181 | 10376 |
| 10392 | 10394 | 10415 | 10452 | 10467 | 10471 | 10483 | 10485 | 10492 | 10496 | 10522 | 10549 | 10576 |
| 10603 | 10714 | 10721 | 10738 | | | | | | | | | |

\$F\$IF = 000110

| | | | | | | | | | | | | |
|-------|------|------|------|------|------|------|------|------|------|------|------|------|
| 8612# | 8906 | 8908 | 8910 | 8941 | 8944 | 8992 | 8994 | 8999 | 9002 | 9009 | 9012 | 9020 |
| 9023 | 9032 | 9035 | 9055 | 9058 | 9065 | 9068 | 9076 | 9079 | 9088 | 9091 | 9112 | 9115 |

| | | | | | | | | | | | | |
|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 9122 | 9125 | 9133 | 9136 | 9145 | 9148 | 9165 | 9172 | 9192 | 9194 | 9214 | 9218 | 9231 |
| 9237 | 9242 | 9245 | 9246 | 9247 | 9249 | 9254 | 9255 | 9257 | 9281 | 9283 | 9296 | 9299 |
| 9307 | 9310 | 9325 | 9327 | 9339 | 9343 | 9350 | 9354 | 9368 | 9370 | 9382 | 9386 | 9395 |
| 9399 | 9413 | 9414 | 9416 | 9418 | 9419 | 9421 | 9435 | 9438 | 9449 | 9455 | 9460 | 9468 |
| 9474 | 9480 | 9493 | 9496 | 9498 | 9505 | 9507 | 9511 | 9559 | 9562 | 9568 | 9571 | 9606 |
| 9608 | 9635 | 9638 | 9643 | 9646 | 9667 | 9670 | 9689 | 9692 | 9701 | 9705 | 9711 | 9715 |
| 9720 | 9723 | 9727 | 9728 | 9729 | 9733 | 9734 | 9738 | 9739 | 9744 | 9758 | 9760 | 9792 |
| 9794 | 9796 | 9814 | 9815 | 9816 | 9819 | 9822 | 9825 | 9828 | 9829 | 9832 | 9833 | 9854 |
| 9857 | 9879 | 9884 | 9886 | 9888 | 9891 | 9892 | 9918 | 9920 | 9921 | 9924 | 9925 | 9935 |
| 9938 | 9940 | 9943 | 9944 | 9946 | 9947 | 9949 | 9951 | 9954 | 9955 | 9956 | 9958 | 9959 |
| 9960 | 9968 | 9970 | 9971 | 9973 | 9974 | 9978 | 9979 | 9984 | 9993 | 9996 | 10009 | 10012 |
| 10015 | 10018 | 10034 | 10035 | 10037 | 10038 | 10041 | 10042 | 10050 | 10052 | 10055 | 10103 | 10116 |
| 10119 | 10120 | 10126 | 10128 | 10130 | 10131 | 10181 | 10188 | 10376 | 10385 | 10387 | 10392 | 10394 |
| 10396 | 10397 | 10415 | 10418 | 10421 | 10452 | 10467 | 10469 | 10471 | 10473 | 10480 | 10483 | 10485 |
| 10488 | 10489 | 10492 | 10495 | 10496 | 10499 | 10522 | 10527 | 10530 | 10549 | 10554 | 10557 | 10576 |
| 10581 | 10584 | 10603 | 10608 | 10611 | 10714 | 10716 | 10718 | 10721 | 10724 | 10738 | 10740 | |
| 8612# | 9695 | 9745 | 10771 | 10772 | 10773 | 10774 | | | | | | |
| 8612# | 9197 | 9258 | 9259 | 10712 | 10726 | 10732 | | | | | | |
| 8612# | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 | 9122 |
| 9133 | 9145 | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 | 9395 |
| 9413 | 9414 | 9449 | 9460 | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 | 9727 |
| 9728 | 9729 | 9739 | 9758 | 9792 | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 | 9886 |
| 9918 | 9921 | 9944 | 9947 | 9949 | 9951 | 9956 | 9968 | 9971 | 9974 | 9979 | 10034 | 10035 |
| 10038 | 10052 | 10103 | 10126 | 10128 | 10181 | 10376 | 10392 | 10394 | 10415 | 10452 | 10467 | 10483 |
| 10485 | 10492 | 10522 | 10549 | 10576 | 10603 | 10714 | | | | | | |
| 8612# | 8906 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 |
| 9122 | 9133 | 9145 | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 |
| 9395 | 9413 | 9414 | 9449 | 9460 | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 |
| 9727 | 9728 | 9729 | 9739 | 9758 | 9792 | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 |
| 9886 | 9918 | 9921 | 9940 | 9944 | 9947 | 9949 | 9951 | 9956 | 9968 | 9971 | 9974 | 9979 |
| 10015 | 10034 | 10035 | 10038 | 10052 | 10103 | 10126 | 10128 | 10181 | 10376 | 10392 | 10394 | 10415 |
| 10452 | 10467 | 10471 | 10483 | 10485 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 | 10714 | 10721 |
| 10738 | | | | | | | | | | | | |
| 8612# | 10704 | 10748 | 10768 | 10776 | 11042 | 11055 | | | | | | |
| 8612# | | | | | | | | | | | | |
| 8612# | 8934 | 8948 | 9807 | 9808 | 10079 | 10081 | 10089 | 10093 | 10102 | 10132 | 10349 | 10351 |
| 10356 | 10363 | 10375 | 10398 | 10414 | 10423 | 10430 | 10431 | 10435 | 10437 | 10451 | 10466 | 10479 |
| 10490 | | | | | | | | | | | | |
| 8612# | 9281 | 9325 | 9368 | 9606 | 9689 | 9739 | 9758 | 9854 | 9879 | 9918 | | |
| 8612# | 8906 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 |
| 9122 | 9133 | 9145 | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 |
| 9395 | 9413 | 9414 | 9449 | 9460 | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 |
| 9727 | 9728 | 9729 | 9739 | 9758 | 9792 | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 |
| 9886 | 9918 | 9921 | 9940 | 9944 | 9947 | 9949 | 9951 | 9956 | 9968 | 9971 | 9974 | 9979 |
| 10015 | 10034 | 10035 | 10038 | 10052 | 10103 | 10126 | 10128 | 10181 | 10376 | 10392 | 10394 | 10415 |
| 10452 | 10467 | 10471 | 10483 | 10485 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 | 10714 | 10721 |
| 10738 | | | | | | | | | | | | |
| 8857# | | | | | | | | | | | | |
| 8857# | | | | | | | | | | | | |
| 11063# | | | | | | | | | | | | |
| 11069# | 11087 | | | | | | | | | | | |
| 8721 | | | | | | | | | | | | |
| 8856# | | | | | | | | | | | | |
| 8857# | 11078* | | | | | | | | | | | |
| 8612# | 8906# | 8910# | 8941# | 8944# | 8992# | 8994# | 8999# | 9002# | 9009# | 9012# | 9020# | 9023# |

SF\$INC= 000210
 SF\$LOO= 000200
 SF\$NAM= 000160
 SF\$NO = 000403

 SF\$OR = 000320

 SF\$RTN= 000300
 SF\$SEL= 000140
 SF\$UNT= 000130

 SF\$WHI= 000120
 SF\$YES= 000402

 \$GDADR 001120
 \$GDDAT 001124
 \$GET42 014440
 \$GTSWR 015036
 \$HD = 000000
 \$HIBTS 001000
 \$ICNT 001104
 \$IFLEV= 177777

| | | | | | | | | | | | | |
|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 9032# | 9035# | 9055# | 9058# | 9065# | 9068# | 9076# | 9079# | 9088# | 9091# | 9112# | 9115# | 9122# |
| 9125# | 9133# | 9136# | 9145# | 9148# | 9165# | 9172# | 9192# | 9194# | 9214# | 9218# | 9231# | 9242# |
| 9245# | 9246# | 9247# | 9249# | 9254# | 9257# | 9281# | 9283# | 9296# | 9299# | 9307# | 9310# | 9325# |
| 9327# | 9339# | 9343# | 9350# | 9354# | 9368# | 9370# | 9382# | 9386# | 9395# | 9399# | 9413# | 9414# |
| 9418# | 9421# | 9435# | 9438# | 9449# | 9455# | 9460# | 9468# | 9474# | 9480# | 9493# | 9496# | 9498# |
| 9505# | 9507# | 9511# | 9559# | 9562# | 9568# | 9571# | 9606# | 9608# | 9635# | 9638# | 9643# | 9646# |
| 9667# | 9670# | 9689# | 9692# | 9701# | 9705# | 9711# | 9715# | 9720# | 9723# | 9727# | 9728# | 9729# |
| 9733# | 9738# | 9739 | 9744# | 9758# | 9760# | 9792# | 9796# | 9814# | 9815# | 9816# | 9819 | 9822 |
| 9828# | 9832# | 9833# | 9854# | 9857# | 9879# | 9886# | 9891# | 9892# | 9918# | 9921# | 9924# | 9925# |
| 9935# | 9938# | 9940# | 9943# | 9944# | 9946# | 9947# | 9949# | 9951# | 9954# | 9956# | 9958# | 9959# |
| 9960# | 9968# | 9970# | 9971# | 9973# | 9974# | 9978# | 9979# | 9984# | 9993# | 9996# | 10009# | 10012# |
| 10015# | 10018# | 10034# | 10035# | 10038# | 10041# | 10042# | 10050# | 10052# | 10055# | 10103# | 10116# | 10119# |
| 10120# | 10126# | 10128# | 10130# | 10131# | 10181# | 10188# | 10376# | 10387# | 10392# | 10394# | 10396# | 10397# |
| 10415# | 10421# | 10452# | 10467# | 10469# | 10471# | 10473# | 10480# | 10483# | 10485# | 10488# | 10489# | 10492# |
| 10495# | 10496# | 10499# | 10522# | 10530# | 10549# | 10557# | 10576# | 10584# | 10603# | 10611# | 10714# | 10718# |
| 10721# | 10724# | 10738# | 10740# | | | | | | | | | |
| 8857# | 11069 | | | | | | | | | | | |
| 8906# | 8910 | 8941# | 8944 | 8992# | 8994 | 8999# | 9002 | 9009# | 9012 | 9020# | 9023 | 9032# |
| 9035 | 9055# | 9058 | 9065# | 9068 | 9076# | 9079 | 9088# | 9091 | 9112# | 9115 | 9122# | 9125 |
| 9133# | 9136 | 9145# | 9148 | 9165# | 9172 | 9192# | 9194 | 9214# | 9218 | 9231# | 9246 | 9247# |
| 9257 | 9281# | 9283 | 9296# | 9299 | 9307# | 9310 | 9325# | 9327 | 9339# | 9343 | 9350# | 9354 |
| 9368# | 9370 | 9382# | 9386 | 9395# | 9399 | 9413# | 9421 | 9435# | 9438 | 9449# | 9455 | 9460# |
| 9468 | 9474# | 9480 | 9493# | 9496 | 9498# | 9505 | 9507# | 9511 | 9559# | 9562 | 9568# | 9571 |
| 9606# | 9608 | 9635# | 9638 | 9643# | 9646 | 9667# | 9670 | 9689# | 9692 | 9701# | 9705 | 9711# |
| 9715 | 9720# | 9723 | 9727# | 9739# | 9744 | 9758# | 9760 | 9792# | 9796 | 9814# | 9833 | 9854# |
| 9857 | 9879# | 9892 | 9918# | 9925 | 9935# | 9938 | 9940# | 9943 | 9944# | 9946 | 9947# | 9960 |
| 9968# | 9970 | 9971# | 9973 | 9974# | 9978 | 9979# | 9984 | 9993# | 9996 | 10009# | 10012 | 10015# |
| 10018 | 10034# | 10050 | 10052# | 10055 | 10103# | 10120 | 10126# | 10131 | 10181# | 10188 | 10376# | 10387 |
| 10392# | 10397 | 10415# | 10421 | 10452# | 10480 | 10483# | 10489 | 10492# | 10495 | 10496# | 10499 | 10522# |
| 10530 | 10549# | 10557 | 10576# | 10584 | 10603# | 10611 | 10714# | 10718 | 10721# | 10724 | 10738# | 10740 |
| 9242# | 9245 | 9249# | 9254 | 9414# | 9418 | 9728# | 9738 | 9815# | 9832 | 9886# | 9891 | 9921# |
| 9924 | 9949# | 9959 | 10035# | 10042 | 10116# | 10119 | 10128# | 10130 | 10394# | 10396 | 10467# | 10469 |
| 10471# | 10473 | 10485# | 10488 | | | | | | | | | |
| 9729# | 9733 | 9816# | 9819# | 9822# | 9828 | 9951# | 9954 | 9956# | 9958 | 10038# | 10041 | |
| 8857# | 11046 | 11075* | | | | | | | | | | |
| 8857# | 11066 | 11069 | 11075 | | | | | | | | | |
| 11072#* | | | | | | | | | | | | |
| 8612# | 8906 | 8908 | 8910 | 8934 | 8941 | 8944 | 8948 | 8992 | 8994 | 8999 | 9002 | 9009 |
| 9012 | 9020 | 9023 | 9032 | 9035 | 9055 | 9058 | 9065 | 9068 | 9076 | 9079 | 9088 | 9091 |
| 9112 | 9115 | 9122 | 9125 | 9133 | 9136 | 9145 | 9148 | 9165 | 9172 | 9192 | 9194 | 9197 |
| 9214 | 9218 | 9231 | 9237 | 9242 | 9245 | 9246 | 9247 | 9249 | 9254 | 9255 | 9257 | 9258 |
| 9259 | 9281 | 9283 | 9296 | 9299 | 9307 | 9310 | 9325 | 9327 | 9339 | 9343 | 9350 | 9354 |
| 9368 | 9370 | 9382 | 9386 | 9395 | 9399 | 9413 | 9414 | 9416 | 9418 | 9419 | 9421 | 9435 |
| 9438 | 9449 | 9455 | 9460 | 9468 | 9474 | 9480 | 9493 | 9496 | 9498 | 9505 | 9507 | 9511 |
| 9559 | 9562 | 9568 | 9571 | 9606 | 9608 | 9635 | 9638 | 9643 | 9646 | 9667 | 9670 | 9689 |
| 9692 | 9695 | 9701 | 9705 | 9711 | 9715 | 9720 | 9723 | 9727 | 9728 | 9729 | 9733 | 9734 |
| 9738 | 9739 | 9744 | 9745 | 9758 | 9760 | 9792 | 9794 | 9796 | 9807 | 9808 | 9814 | 9815 |
| 9816 | 9819 | 9822 | 9825 | 9828 | 9829 | 9832 | 9833 | 9854 | 9857 | 9879 | 9884 | 9886 |
| 9888 | 9891 | 9892 | 9918 | 9920 | 9921 | 9924 | 9925 | 9935 | 9938 | 9940 | 9943 | 9944 |
| 9946 | 9947 | 9949 | 9951 | 9954 | 9955 | 9956 | 9958 | 9959 | 9960 | 9968 | 9970 | 9971 |
| 9973 | 9974 | 9978 | 9979 | 9984 | 9993 | 9996 | 10009 | 10012 | 10015 | 10018 | 10034 | 10035 |
| 10037 | 10038 | 10041 | 10042 | 10050 | 10052 | 10055 | 10079 | 10081 | 10089 | 10093 | 10102 | 10103 |
| 10116 | 10119 | 10120 | 10126 | 10128 | 10130 | 10131 | 10132 | 10181 | 10188 | 10349 | 10351 | 10356 |
| 10363 | 10375 | 10376 | 10385 | 10387 | 10392 | 10394 | 10396 | 10397 | 10398 | 10414 | 10415 | 10418 |
| 10421 | 10423 | 10430 | 10431 | 10435 | 10437 | 10451 | 10452 | 10466 | 10467 | 10469 | 10471 | 10473 |
| 10479 | 10480 | 10483 | 10485 | 10488 | 10489 | 10490 | 10492 | 10495 | 10496 | 10499 | 10522 | 10527 |

\$INTAG 001135
\$ISKO = 000001

\$ISK1 = 000001

\$ISK2 = 000001
\$ITEMB 001114
\$LF 001172
\$LFLG 016005
\$LOCTA= 177777

\$LPADR 001106
 \$LPERR 001110

\$LSTCN= 177777

\$LSTIN= 000000

| | | | | | | | | | | | | |
|-------|-------|--------|-------|-------|--------|-------|-------|-------|-------|-------|-------|-------|
| 10530 | 10549 | 10554 | 10557 | 10576 | 10581 | 10584 | 10603 | 10608 | 10611 | 10704 | 10712 | 10714 |
| 10716 | 10718 | 10721 | 10724 | 10726 | 10732 | 10738 | 10739 | 10740 | 10741 | 10748 | 10768 | 10771 |
| 10772 | 10773 | 10774 | 10776 | 11042 | 11055 | | | | | | | |
| 8857# | 8871* | 11078* | | | | | | | | | | |
| 8857# | 8871* | 8935* | 8997* | 9006* | 9016* | 9027* | 9053* | 9062* | 9072* | 9083* | 9110* | 9119* |
| 9129* | 9140* | 9163* | 9285* | 9303* | 9329* | 9372* | 9425* | 9459* | 9471* | 9483* | 9543* | 9574* |
| 9614* | 9655* | 9926* | 9986* | 11075 | 11078* | | | | | | | |
| 8612# | 8615# | 8906 | 8908 | 8910 | 8934 | 8941 | 8944 | 8948 | 8992 | 8994 | 8999 | 9002 |
| 9009 | 9012 | 9020 | 9023 | 9032 | 9035 | 9055 | 9058 | 9065 | 9068 | 9076 | 9079 | 9088 |
| 9091 | 9112 | 9115 | 9122 | 9125 | 9133 | 9136 | 9145 | 9148 | 9165 | 9172 | 9192 | 9194 |
| 9197 | 9214 | 9218 | 9231 | 9237 | 9242 | 9245 | 9246 | 9247 | 9249 | 9254 | 9255 | 9257 |
| 9259 | 9281 | 9283 | 9296 | 9299 | 9307 | 9310 | 9325 | 9327 | 9339 | 9343 | 9350 | 9354 |
| 9368 | 9370 | 9382 | 9386 | 9395 | 9399 | 9413 | 9414 | 9416 | 9418 | 9419 | 9421 | 9435 |
| 9438 | 9449 | 9455 | 9460 | 9469 | 9474 | 9480 | 9493 | 9496 | 9498 | 9505 | 9507 | 9511 |
| 9559 | 9562 | 9568 | 9571 | 9606 | 9608 | 9635 | 9638 | 9643 | 9646 | 9667 | 9670 | 9689 |
| 9692 | 9695 | 9701 | 9705 | 9711 | 9715 | 9720 | 9723 | 9727 | 9728 | 9729 | 9733 | 9734 |
| 9738 | 9739 | 9744 | 9745 | 9758 | 9760 | 9792 | 9794 | 9796 | 9807 | 9808 | 9814 | 9815 |
| 9816 | 9819 | 9822 | 9825 | 9828 | 9829 | 9832 | 9833 | 9854 | 9857 | 9879 | 9884 | 9886 |
| 9888 | 9891 | 9892 | 9918 | 9920 | 9921 | 9924 | 9925 | 9935 | 9938 | 9940 | 9943 | 9944 |
| 9946 | 9947 | 9949 | 9951 | 9954 | 9955 | 9956 | 9958 | 9959 | 9960 | 9968 | 9970 | 9971 |
| 9973 | 9974 | 9978 | 9979 | 9984 | 9993 | 9996 | 10009 | 10012 | 10015 | 10018 | 10034 | 10035 |
| 10037 | 10038 | 10041 | 10042 | 10050 | 10052 | 10055 | 10079 | 10081 | 10089 | 10093 | 10102 | 10103 |
| 10116 | 10119 | 10120 | 10126 | 10128 | 10130 | 10131 | 10132 | 10181 | 10188 | 10349 | 10351 | 10356 |
| 10363 | 10375 | 10376 | 10385 | 10387 | 10392 | 10394 | 10396 | 10397 | 10398 | 10414 | 10415 | 10418 |
| 10421 | 10423 | 10430 | 10431 | 10435 | 10437 | 10451 | 10452 | 10466 | 10467 | 10469 | 10471 | 10473 |
| 10479 | 10480 | 10483 | 10485 | 10488 | 10489 | 10490 | 10492 | 10495 | 10496 | 10499 | 10522 | 10527 |
| 10530 | 10549 | 10554 | 10557 | 10576 | 10581 | 10584 | 10603 | 10608 | 10611 | 10704 | 10712 | 10714 |
| 10716 | 10718 | 10721 | 10724 | 10732 | 10738 | 10740 | 10748 | 10768 | 10771 | 10772 | 10773 | 10774 |
| 10776 | 11042 | 11055 | | | | | | | | | | |
| 8612# | 8613# | 8892 | 8894 | 8896 | 8897 | 8898 | 8899 | 8900 | 8905 | 8906 | 8908 | 8909 |
| 8928 | 8933 | 8935 | 8937 | 8941 | 8946 | 8947 | 8948 | 8992 | 8997 | 8999 | 9006 | 9007 |
| 9009 | 9016 | 9018 | 9020 | 9027 | 9029 | 9032 | 9053 | 9055 | 9062 | 9063 | 9065 | 9072 |
| 9074 | 9076 | 9083 | 9085 | 9088 | 9110 | 9112 | 9119 | 9120 | 9122 | 9129 | 9131 | 9133 |
| 9140 | 9142 | 9145 | 9163 | 9165 | 9192 | 9196 | 9199 | 9200 | 9207 | 9211 | 9214 | 9227 |
| 9231 | 9234 | 9237 | 9241 | 9242 | 9247 | 9248 | 9249 | 9253 | 9255 | 9256 | 9258 | 9259 |
| 9281 | 9285 | 9288 | 9293 | 9296 | 9303 | 9307 | 9325 | 9329 | 9333 | 9337 | 9339 | 9349 |
| 9350 | 9368 | 9372 | 9376 | 9380 | 9382 | 9393 | 9395 | 9413 | 9414 | 9416 | 9419 | 9425 |
| 9430 | 9432 | 9435 | 9441 | 9443 | 9446 | 9449 | 9459 | 9460 | 9471 | 9474 | 9483 | 9489 |
| 9490 | 9493 | 9498 | 9507 | 9534 | 9537 | 9539 | 9543 | 9545 | 9548 | 9554 | 9556 | 9559 |
| 9565 | 9568 | 9574 | 9576 | 9578 | 9581 | 9583 | 9606 | 9612 | 9614 | 9615 | 9617 | 9623 |
| 9627 | 9629 | 9632 | 9635 | 9640 | 9643 | 9655 | 9658 | 9660 | 9662 | 9665 | 9667 | 9689 |
| 9695 | 9700 | 9701 | 9708 | 9710 | 9711 | 9718 | 9720 | 9727 | 9728 | 9729 | 9734 | 9739 |
| 9745 | 9758 | 9780 | 9782 | 9783 | 9786 | 9787 | 9790 | 9791 | 9792 | 9793 | 9794 | 9795 |
| 9801 | 9803 | 9808 | 9812 | 9813 | 9814 | 9815 | 9816 | 9819 | 9822 | 9825 | 9829 | 9854 |
| 9856 | 9859 | 9877 | 9879 | 9881 | 9882 | 9883 | 9884 | 9885 | 9886 | 9887 | 9888 | 9890 |
| 9918 | 9920 | 9921 | 9926 | 9928 | 9930 | 9932 | 9934 | 9935 | 9939 | 9940 | 9942 | 9944 |
| 9945 | 9947 | 9949 | 9951 | 9953 | 9955 | 9956 | 9957 | 9968 | 9971 | 9974 | 9979 | 9986 |
| 9988 | 9990 | 9993 | 9999 | 10000 | 10004 | 10006 | 10008 | 10009 | 10015 | 10034 | 10035 | 10037 |
| 10038 | 10040 | 10052 | 10053 | 10054 | 10078 | 10080 | 10081 | 10087 | 10088 | 10090 | 10091 | 10092 |
| 10093 | 10099 | 10100 | 10101 | 10103 | 10105 | 10106 | 10108 | 10110 | 10112 | 10114 | 10116 | 10122 |
| 10124 | 10125 | 10126 | 10127 | 10128 | 10129 | 10132 | 10144 | 10152 | 10154 | 10155 | 10157 | 10181 |
| 10208 | 10210 | 10211 | 10213 | 10219 | 10221 | 10222 | 10224 | 10229 | 10231 | 10232 | 10234 | 10239 |
| 10241 | 10242 | 10244 | 10249 | 10251 | 10252 | 10254 | 10259 | 10261 | 10262 | 10264 | 10269 | 10271 |
| 10272 | 10274 | 10279 | 10281 | 10282 | 10284 | 10344 | 10348 | 10350 | 10351 | 10354 | 10355 | 10357 |
| 10358 | 10359 | 10360 | 10361 | 10362 | 10363 | 10369 | 10371 | 10372 | 10374 | 10376 | 10378 | 10380 |
| 10381 | 10383 | 10384 | 10385 | 10386 | 10388 | 10391 | 10392 | 10393 | 10394 | 10395 | 10398 | 10401 |

| | 10485# | 10488 | 10714# | 10716 | 10718 | 10721# | 10724 | 10772# | 10773 | | | | |
|------------------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| \$NSK3 = 000110 | 9729# | 9733 | 10467# | 10469 | 10471# | 10473 | | | | | | | |
| \$NULL 001154 | 8857# | 11066 | | | | | | | | | | | |
| \$NWTST= 000001 | 8923# | 8986# | 9044# | 9104# | 9157# | 9186# | 9275# | 9319# | 9362# | 9407# | 9528# | 9601# | 9684# |
| | 9753# | 9910# | 10029# | 10070# | 10338# | | | | | | | | |
| \$SOCNT 017136 | 11084#* | | | | | | | | | | | | |
| \$SOMODE 017140 | 11084#* | | | | | | | | | | | | |
| \$OVER 016452 | 11078# | | | | | | | | | | | | |
| \$PASS 001202 | 8857# | 8871* | 11063* | 11078 | | | | | | | | | |
| \$PASTM 001006 | 8856# | | | | | | | | | | | | |
| \$QUES 001170 | 8857# | 11066 | 11069 | 11075 | | | | | | | | | |
| \$RDCHR 015250 | 11069# | 11087 | | | | | | | | | | | |
| \$RDDEC= ***** U | 11087 | | | | | | | | | | | | |
| \$RDLIN 015370 | 11069# | 11087 | | | | | | | | | | | |
| \$RDOCT= ***** U | 11087 | | | | | | | | | | | | |
| \$RDSZ = 000010 | 11069# | | | | | | | | | | | | |
| \$RTNAD 014462 | 11063# | | | | | | | | | | | | |
| \$R2A = ***** U | 11087 | | | | | | | | | | | | |
| \$SAVLE= 177777 | 8612# | 9259# | 9695# | 10732# | 10771# | 10772# | | | | | | | |
| \$SAVRE= ***** U | 11087 | | | | | | | | | | | | |
| \$SCOPE 016210 | 8871 | 11078# | | | | | | | | | | | |
| \$SETUP= 000127 | 8868# | 8871 | 8872 | 11063 | 11069 | 11075 | 11078 | | | | | | |
| \$SSKO = 000253 | 9259# | 9695# | 10732# | 10771# | 10772# | | | | | | | | |
| \$STUP = 177777 | 8868# | | | | | | | | | | | | |
| \$SVLAD 016416 | 11078# | | | | | | | | | | | | |
| \$SVPC = 000430 | 8854# | | | | | | | | | | | | |
| \$SWR = 167400 | 8617# | 8721 | 8723 | 8857 | 8871 | 8923 | 8986 | 9044 | 9104 | 9157 | 9186 | 9275 | 9319 |
| | 9362 | 9407 | 9528 | 9601 | 9684 | 9753 | 9910 | 10029 | 10070 | 10338 | 11063 | 11075 | 11078 |
| \$SWREG 001216 | 8857# | 8871 | | | | | | | | | | | |
| \$SWRMK= 000000 | 8723 | 11078 | | | | | | | | | | | |
| \$TAGLE= 177777 | 8612# | 8906# | 8908# | 8910# | 8934# | 8941# | 8944# | 8948# | 8992# | 8994# | 8999# | 9002# | 9009# |
| | 9012# | 9020# | 9023# | 9032# | 9035# | 9055# | 9058# | 9065# | 9068# | 9076# | 9079# | 9088# | 9091# |
| | 9112# | 9115# | 9122# | 9125# | 9133# | 9136# | 9145# | 9148# | 9165# | 9172# | 9192# | 9194# | 9197# |
| | 9214# | 9218# | 9231# | 9237# | 9242# | 9245# | 9246# | 9247# | 9249# | 9254# | 9255# | 9257# | 9258 |
| | 9259# | 9281# | 9283# | 9296# | 9299# | 9307# | 9310# | 9325# | 9327# | 9339# | 9343# | 9350# | 9354# |
| | 9368# | 9370# | 9382# | 9386# | 9395# | 9399# | 9413# | 9414# | 9416# | 9418# | 9419# | 9421# | 9435# |
| | 9438# | 9449# | 9455# | 9460# | 9468# | 9474# | 9480# | 9493# | 9496# | 9498# | 9505# | 9507# | 9511# |
| | 9559# | 9562# | 9568# | 9571# | 9606# | 9608# | 9635# | 9638# | 9643# | 9646# | 9667# | 9670# | 9689# |
| | 9692# | 9695# | 9701# | 9705# | 9711# | 9715# | 9720# | 9723# | 9727# | 9728# | 9729# | 9733# | 9734# |
| | 9738# | 9739# | 9744# | 9745# | 9758# | 9760# | 9792# | 9794# | 9796# | 9807# | 9808# | 9814# | 9815# |
| | 9816# | 9819# | 9822# | 9825# | 9828# | 9829# | 9832# | 9833# | 9854# | 9857# | 9879# | 9884# | 9886# |
| | 9888# | 9891# | 9892# | 9918# | 9920# | 9921# | 9924# | 9925# | 9935# | 9938# | 9940# | 9943# | 9944# |
| | 9946# | 9947# | 9949# | 9951# | 9954# | 9955# | 9956# | 9958# | 9959# | 9960# | 9968# | 9970# | 9971# |
| | 9973# | 9974# | 9978# | 9979# | 9984# | 9993# | 9996# | 10009# | 10012# | 10015# | 10018# | 10034# | 10035# |
| | 10037# | 10038# | 10041# | 10042# | 10050# | 10052# | 10055# | 10079# | 10081# | 10089# | 10093# | 10102# | 10103# |
| | 10116# | 10119# | 10120# | 10126# | 10128# | 10130# | 10131# | 10132# | 10181# | 10188# | 10349# | 10351# | 10356# |
| | 10363# | 10375# | 10376# | 10385# | 10387# | 10392# | 10394# | 10396# | 10397# | 10398# | 10414# | 10415# | 10418# |
| | 10421# | 10423# | 10430# | 10431# | 10435# | 10437# | 10451# | 10452# | 10466# | 10467# | 10469# | 10471# | 10473# |
| | 10479# | 10480# | 10483# | 10485# | 10488# | 10489# | 10490# | 10492# | 10495# | 10496# | 10499# | 10522# | 10527# |
| | 10530# | 10549# | 10554# | 10557# | 10576# | 10581# | 10584# | 10603# | 10608# | 10611# | 10712# | 10714# | 10716# |
| | 10718# | 10721# | 10724# | 10726 | 10732# | 10738# | 10740# | 10771# | 10772# | 10773# | 10774# | | |
| \$TAGNU= 000257 | 8612# | 8906# | 8908# | 8934# | 8941# | 8992# | 8999# | 9009# | 9020# | 9032# | 9055# | 9065# | 9076# |
| | 9088# | 9112# | 9122# | 9133# | 9145# | 9165# | 9192# | 9197# | 9214# | 9231# | 9237# | 9242# | 9247# |
| | 9249# | 9255# | 9281# | 9296# | 9307# | 9325# | 9339# | 9350# | 9368# | 9382# | 9395# | 9413# | 9414# |
| | 9416# | 9419# | 9435# | 9449# | 9460# | 9474# | 9493# | 9498# | 9507# | 9559# | 9568# | 9606# | 9635# |
| | 9643# | 9667# | 9689# | 9695# | 9701# | 9711# | 9720# | 9727# | 9728# | 9729# | 9734# | 9739# | 9758# |

\$TEMP = 000300

| | | | | | | | | | | | | |
|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 9792# | 9794# | 9807# | 9808# | 9814# | 9815# | 9816# | 9819# | 9822# | 9825# | 9829# | 9854# | 9879# |
| 9884# | 9886# | 9888# | 9918# | 9920# | 9921# | 9935# | 9940# | 9944# | 9947# | 9949# | 9951# | 9955# |
| 9956# | 9968# | 9971# | 9974# | 9979# | 9993# | 10009# | 10015# | 10034# | 10035# | 10037# | 10038# | 10052# |
| 10079# | 10089# | 10102# | 10103# | 10116# | 10126# | 10128# | 10181# | 10349# | 10356# | 10375# | 10376# | 10385# |
| 10392# | 10394# | 10414# | 10415# | 10418# | 10430# | 10435# | 10451# | 10452# | 10466# | 10467# | 10471# | 10479# |
| 10483# | 10485# | 10490# | 10492# | 10496# | 10522# | 10527# | 10549# | 10554# | 10576# | 10581# | 10603# | 10608# |
| 10704# | 10712# | 10714# | 10716# | 10721# | 10738# | 10768# | 10771# | 10772# | 11042# | | | |
| 8892# | 8894# | 8896# | 8897# | 8898# | 8899# | 8900# | 8905# | 8908# | 8909# | 8910# | 8928# | 8933# |
| 8935# | 8937# | 8944# | 8946# | 8947# | 8948# | 8994# | 8997# | 9002# | 9006# | 9007# | 9012# | 9016# |
| 9018# | 9023# | 9027# | 9029# | 9035# | 9053# | 9058# | 9062# | 9063# | 9068# | 9072# | 9074# | 9079# |
| 9083# | 9085# | 9091# | 9110# | 9115# | 9119# | 9120# | 9125# | 9129# | 9131# | 9136# | 9140# | 9142# |
| 9148# | 9163# | 9172# | 9194# | 9196# | 9199# | 9200# | 9207# | 9218# | 9227# | 9234# | 9237# | 9245# |
| 9246# | 9248# | 9253# | 9254# | 9255# | 9256# | 9257# | 9258# | 9259# | 9283# | 9285# | 9288# | 9299# |
| 9303# | 9310# | 9327# | 9329# | 9333# | 9343# | 9349# | 9354# | 9370# | 9372# | 9376# | 9386# | 9393# |
| 9399# | 9416# | 9418# | 9419# | 9421# | 9425# | 9430# | 9438# | 9441# | 9446# | 9455# | 9459# | 9468# |
| 9471# | 9480# | 9483# | 9489# | 9496# | 9505# | 9511# | 9534# | 9537# | 9539# | 9543# | 9548# | 9554# |
| 9562# | 9571# | 9574# | 9576# | 9578# | 9581# | 9608# | 9612# | 9614# | 9615# | 9617# | 9623# | 9629# |
| 9638# | 9646# | 9655# | 9658# | 9660# | 9662# | 9670# | 9692# | 9695# | 9705# | 9708# | 9715# | 9718# |
| 9723# | 9733# | 9734# | 9738# | 9739# | 9744# | 9745# | 9760# | 9780# | 9782# | 9783# | 9786# | 9787# |
| 9790# | 9791# | 9793# | 9794# | 9795# | 9796# | 9801# | 9803# | 9808# | 9812# | 9813# | 9819# | 9822# |
| 9825# | 9828# | 9829# | 9832# | 9833# | 9856# | 9857# | 9859# | 9877# | 9881# | 9882# | 9883# | 9884# |
| 9885# | 9887# | 9888# | 9890# | 9891# | 9892# | 9920# | 9924# | 9925# | 9926# | 9928# | 9930# | 9932# |
| 9938# | 9939# | 9942# | 9943# | 9945# | 9946# | 9953# | 9954# | 9955# | 9957# | 9958# | 9959# | 9960# |
| 9970# | 9973# | 9978# | 9984# | 9986# | 9988# | 9996# | 9999# | 10004# | 10006# | 10012# | 10018# | 10037# |
| 10040# | 10041# | 10042# | 10050# | 10053# | 10054# | 10055# | 10078# | 10080# | 10081# | 10087# | 10088# | 10090# |
| 10091# | 10092# | 10093# | 10099# | 10100# | 10101# | 10105# | 10106# | 10108# | 10110# | 10112# | 10119# | 10120# |
| 10122# | 10124# | 10125# | 10129# | 10130# | 10131# | 10132# | 10144# | 10152# | 10154# | 10155# | 10157# | 10188# |
| 10208# | 10210# | 10211# | 10213# | 10219# | 10221# | 10222# | 10224# | 10229# | 10231# | 10232# | 10234# | 10239# |
| 10241# | 10242# | 10244# | 10249# | 10251# | 10252# | 10254# | 10259# | 10261# | 10262# | 10264# | 10269# | 10271# |
| 10272# | 10274# | 10279# | 10281# | 10282# | 10284# | 10344# | 10348# | 10350# | 10351# | 10354# | 10355# | 10357# |
| 10358# | 10359# | 10360# | 10361# | 10362# | 10363# | 10369# | 10371# | 10372# | 10374# | 10378# | 10380# | 10381# |
| 10383# | 10384# | 10385# | 10386# | 10387# | 10388# | 10391# | 10395# | 10396# | 10397# | 10398# | 10401# | 10402# |
| 10403# | 10404# | 10409# | 10411# | 10413# | 10416# | 10417# | 10418# | 10419# | 10420# | 10421# | 10422# | 10423# |
| 10426# | 10431# | 10434# | 10436# | 10437# | 10444# | 10445# | 10446# | 10447# | 10449# | 10450# | 10453# | 10455# |
| 10457# | 10459# | 10461# | 10462# | 10464# | 10468# | 10469# | 10472# | 10473# | 10475# | 10477# | 10479# | 10480# |
| 10481# | 10482# | 10487# | 10488# | 10489# | 10490# | 10495# | 10499# | 10519# | 10520# | 10521# | 10524# | 10526# |
| 10527# | 10529# | 10530# | 10534# | 10535# | 10536# | 10537# | 10539# | 10542# | 10546# | 10547# | 10548# | 10551# |
| 10553# | 10554# | 10556# | 10557# | 10561# | 10562# | 10563# | 10564# | 10566# | 10569# | 10573# | 10574# | 10575# |
| 10578# | 10580# | 10581# | 10583# | 10584# | 10588# | 10589# | 10590# | 10591# | 10593# | 10596# | 10600# | 10601# |
| 10602# | 10605# | 10607# | 10608# | 10610# | 10611# | 10615# | 10616# | 10617# | 10618# | 10620# | 10623# | 10705# |
| 10706# | 10707# | 10715# | 10716# | 10717# | 10718# | 10723# | 10724# | 10726# | 10731# | 10732# | 10740# | 10748# |
| 10770# | 10771# | 10772# | 10773# | 10774# | 10776# | 10776# | 10931# | 11055# | | | | |
| 8857# | 8923* | 8986* | 9044* | 9104* | 9157* | 9186* | 9275* | 9319* | 9362* | 9407* | 9528* | 9601* |
| 9684* | 9753* | 9910* | 10070* | 10338* | 11044 | 11078* | | | | | | |
| 8857# | 8871* | 8923* | 8986* | 9044* | 9104* | 9157* | 9186* | 9275* | 9319* | 9362* | 9407* | 9528* |
| 9601* | 9684* | 9753* | 9910* | 10029* | 10070* | 10338* | 11063* | 11078* | | | | |
| 8857# | 11069 | | | | | | | | | | | |
| 8857# | 10903 | 11069* | | | | | | | | | | |
| 8618# | 8721 | 8923# | 8986# | 8993 | 9044# | 9104# | 9157# | 9186# | 9193 | 9260 | 9275# | 9282 |
| 9319# | 9326 | 9362# | 9369 | 9407# | 9417 | 9420 | 9454 | 9467 | 9479 | 9504 | 9513 | 9528# |
| 9601# | 9684# | 9691 | 9704 | 9714 | 9732 | 9737 | 9743 | 9753# | 9834 | 9910# | 10020 | 10029# |
| 10070# | 10190 | 10338# | | | | | | | | | | |
| 8857# | 11066* | | | | | | | | | | | |
| 8857# | 11066 | | | | | | | | | | | |
| 8857# | 11066 | | | | | | | | | | | |
| 8871 | 11087# | | | | | | | | | | | |

\$TESTN 001200
 \$TIMES 001160
 \$TKB 001146
 \$TKS 001144
 \$TN = 000023
 \$TPB 001152
 \$TPFLG 001157
 \$TPS 001150
 \$TRAP 017142

\$STRAP2 017164
\$TRP = 000012
\$TRPAD 017176
\$TSKO = 000251

| | | | | | | | | | | | | | |
|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--|
| 11087# | | | | | | | | | | | | | |
| 11087# | | | | | | | | | | | | | |
| 11087# | | | | | | | | | | | | | |
| 8906# | 8908# | 8910 | 8934# | 8948 | 8992# | 8994 | 8999# | 9002 | 9009# | 9012 | 9020# | 9023 | |
| 9032# | 9035 | 9055# | 9058 | 9065# | 9068 | 9076# | 9079 | 9088# | 9091 | 9112# | 9115 | 9122# | |
| 9125 | 9133# | 9136 | 9145# | 9148 | 9165# | 9172 | 9192# | 9194 | 9197# | 9259 | 9281# | 9283 | |
| 9296# | 9299 | 9307# | 9310 | 9325# | 9327 | 9339# | 9343 | 9350# | 9354 | 9368# | 9370 | 9382# | |
| 9386 | 9395# | 9399 | 9413# | 9419# | 9421 | 9435# | 9438 | 9449# | 9455 | 9460# | 9468 | 9474# | |
| 9480 | 9493# | 9496 | 9498# | 9505 | 9507# | 9511 | 9559# | 9562 | 9568# | 9571 | 9606# | 9608 | |
| 9635# | 9638 | 9643# | 9646 | 9667# | 9670 | 9689# | 9692 | 9695# | 9745 | 9758# | 9760 | 9792# | |
| 9794# | 9796 | 9807# | 9808 | 9814# | 9833 | 9854# | 9857 | 9879# | 9884# | 9892 | 9918# | 9920# | |
| 9925 | 9935# | 9938 | 9940# | 9943 | 9944# | 9946 | 9947# | 9960 | 9968# | 9970 | 9971# | 9973 | |
| 9974# | 9978 | 9979# | 9984 | 9993# | 9996 | 10009# | 10012 | 10015# | 10018 | 10034# | 10050 | 10052# | |
| 10055 | 10079# | 10081 | 10089# | 10093 | 10102# | 10132 | 10181# | 10188 | 10349# | 10351 | 10356# | 10363 | |
| 10375# | 10398 | 10414# | 10423 | 10430# | 10431 | 10435# | 10437 | 10451# | 10490 | 10492# | 10495 | 10496# | |
| 10499 | 10522# | 10527# | 10530 | 10549# | 10554# | 10557 | 10576# | 10581# | 10584 | 10603# | 10608# | 10611 | |
| 10712# | 10732 | 10738# | 10740 | 10771# | 10774 | | | | | | | | |
| 8941# | 8944 | 9197# | 9258 | 9259 | 9414# | 9416# | 9418 | 9695# | 9745 | 9815# | 9829# | 9832 | |
| 9886# | 9888# | 9891 | 9921# | 9924 | 9949# | 9955# | 9959 | 10035# | 10037# | 10042 | 10103# | 10120 | |
| 10126# | 10131 | 10376# | 10385# | 10387 | 10392# | 10397 | 10415# | 10418# | 10421 | 10452# | 10480 | 10483# | |
| 10489 | 10712# | 10726 | 10732 | 10771# | 10774 | | | | | | | | |
| 9214# | 9218 | 9231# | 9237# | 9246 | 9247# | 9255# | 9257 | 9701# | 9705 | 9711# | 9715 | 9720# | |
| 9723 | 9727# | 9739# | 9744 | 9816# | 9819# | 9828 | 9951# | 9954 | 9956# | 9958 | 10038# | 10041 | |
| 10116# | 10119 | 10128# | 10130 | 10394# | 10396 | 10466# | 10479 | 10485# | 10488 | 10714# | 10716# | 10718 | |
| 10721# | 10724 | 10772# | 10773 | | | | | | | | | | |
| 9242# | 9245 | 9249# | 9254 | 9728# | 9734# | 9738 | 9739# | 9744 | 9819# | 9822# | 9828 | 10467# | |
| 10469 | 10471# | 10473 | 10772# | 10773 | | | | | | | | | |
| 9729# | 9733 | 9822# | 9825# | 9828 | | | | | | | | | |

\$TSK1 = 000250

\$TSK2 = 000254

\$TSK3 = 000253

\$TSK4 = 000126
\$STSM 001004
\$STSTM 001102
\$TTYIN 015476
\$STYPBN= ***** U
\$STYPDS 016470
\$STYPE 014504
\$STYPEC 014716
\$STYPEX 014764
\$STYPOC 016740
\$STYPON 016754
\$STYPOS 016714
\$SUNIT 001206
\$SUNITM 001010
\$SUSWR 001220

| | | | | | | | | | | | | | |
|--------|-------|--------|-------|-------|--------|-------|--------|--------|------|------|------|------|--|
| 11087# | | | | | | | | | | | | | |
| 11081# | 11087 | | | | | | | | | | | | |
| 11066# | 11072 | 11087 | | | | | | | | | | | |
| 11066# | 11069 | | | | | | | | | | | | |
| 11066# | | | | | | | | | | | | | |
| 11084# | 11087 | | | | | | | | | | | | |
| 11084# | 11087 | | | | | | | | | | | | |
| 11084# | 11087 | | | | | | | | | | | | |
| 8857# | 8880* | 10054* | 10898 | 10958 | 10962* | 10966 | 10975* | 10986* | | | | | |
| 8857# | | | | | | | | | | | | | |
| 8857# | 8992 | 9281 | 9325 | 9368 | 9414 | 9606 | 9689 | 9728 | 9739 | 9758 | 9792 | 9918 | |
| 9947 | 9949 | 10415 | 10810 | 10814 | 10896 | 10984 | | | | | | | |
| 8857# | 10956 | | | | | | | | | | | | |

\$VECT1 001244
\$VECT2 001246
\$XTSTR 016222
\$YESNO= 000001

| | | | | | | | | | | | | | |
|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--|
| 8897# | 8898# | 8899# | 8905# | 8946# | 8947# | 9007# | 9018# | 9029# | 9063# | 9074# | 9085# | 9120# | |
| 9131# | 9142# | 9248# | 9349# | 9539# | 9548# | 9554# | 9576# | 9617# | 9629# | 9658# | 9695# | 9801# | |
| 9803# | 9812# | 9813# | 9859# | 9882# | 9883# | 9885# | 9887# | 9890# | 9930# | 9942# | 9945# | 9953# | |
| 9957# | 9988# | 9999# | 10040# | 10054# | 10092# | 10105# | 10106# | 10108# | 10112# | 10122# | 10124# | 10125# | |
| 10129# | 10211# | 10213# | 10222# | 10224# | 10232# | 10234# | 10242# | 10244# | 10252# | 10254# | 10262# | 10264# | |
| 10272# | 10274# | 10282# | 10284# | 10359# | 10362# | 10378# | 10380# | 10381# | 10383# | 10384# | 10386# | 10388# | |
| 10391# | 10395# | 10422# | 10436# | 10455# | 10459# | 10462# | 10468# | 10472# | 10475# | 10477# | 10481# | 10482# | |
| 10487# | 10520# | 10524# | 10526# | 10529# | 10535# | 10537# | 10539# | 10542# | 10547# | 10551# | 10553# | 10556# | |
| 10562# | 10564# | 10566# | 10569# | 10574# | 10578# | 10580# | 10583# | 10589# | 10591# | 10593# | 10596# | 10601# | |
| 10605# | 10607# | 10610# | 10616# | 10618# | 10620# | 10623# | 10731# | 10771# | 10772# | 10931# | | | |

| | | | |
|-------|--------|--------|--------|
| \$107 | 005320 | 9739 | 9744# |
| \$11 | 002440 | 9032 | 9035# |
| \$110 | 005320 | 9739 | 9744# |
| \$111 | 005360 | 9758# | |
| \$112 | 005364 | 9758 | 9760# |
| \$113 | 005452 | 9792 | 9794# |
| \$114 | 005460 | 9794 | 9796# |
| \$115 | 005476 | 9807# | 9808 |
| \$116 | 005514 | 9808# | |
| \$117 | 005620 | 9814 | 9833# |
| \$12 | 002510 | 9055 | 9058# |
| \$120 | 005616 | 9815 | 9829# |
| \$121 | 005562 | 9816 | 9819# |
| \$122 | 005614 | 9819 | 9828# |
| \$123 | 005576 | 9819 | 9822# |
| \$124 | 005614 | 9822 | 9828# |
| \$125 | 005612 | 9822 | 9825# |
| \$126 | 005614 | 9825 | 9828# |
| \$127 | 005620 | 9829 | 9832# |
| \$13 | 002536 | 9065 | 9068# |
| \$130 | 005652 | 9854 | 9857# |
| \$131 | 005712 | 9879# | |
| \$132 | 005734 | 9879 | 9884# |
| \$133 | 005766 | 9884 | 9892# |
| \$134 | 005760 | 9886 | 9888# |
| \$135 | 005766 | 9888 | 9891# |
| \$136 | 006030 | 9918# | |
| \$137 | 006036 | 9918 | 9920# |
| \$14 | 002564 | 9076 | 9079# |
| \$140 | 006052 | 9920 | 9925# |
| \$141 | 006052 | 9921 | 9924# |
| \$142 | 006134 | 9935 | 9938# |
| \$143 | 006152 | 9940 | 9943# |
| \$144 | 006166 | 9944 | 9946# |
| \$145 | 006240 | 9947 | 9960# |
| \$146 | 006224 | 9949 | 9955# |
| \$147 | 006222 | 9951 | 9954# |
| \$15 | 002614 | 9088 | 9091# |
| \$150 | 006240 | 9955 | 9959# |
| \$151 | 006240 | 9956 | 9958# |
| \$152 | 006266 | 9968 | 9970# |
| \$153 | 006300 | 9971 | 9973# |
| \$154 | 006312 | 9974 | 9978# |
| \$155 | 006324 | 9979 | 9984# |
| \$156 | 006374 | 9993 | 9996# |
| \$157 | 006464 | 10009 | 10012# |
| \$16 | 002652 | 9112 | 9115# |
| \$160 | 006476 | 10015 | 10018# |
| \$161 | 006750 | 10034 | 10050# |
| \$162 | 006574 | 10035 | 10037# |
| \$163 | 006646 | 10037 | 10042# |
| \$164 | 006646 | 10038 | 10041# |
| \$165 | 006774 | 10052 | 10055# |
| \$166 | 007022 | 10079# | 10081 |
| \$167 | 007054 | 10089# | 10093 |
| \$17 | 002700 | 9122 | 9125# |

| | | | |
|-------|--------|--------|--------|
| \$170 | 007112 | 10102# | 10132 |
| \$171 | 007202 | 10103 | 10120# |
| \$172 | 007202 | 10116 | 10119# |
| \$173 | 007244 | 10126 | 10131# |
| \$174 | 007244 | 10128 | 10130# |
| \$175 | 007422 | 10181 | 10188# |
| \$176 | 010024 | 10349# | 10351 |
| \$177 | 010044 | 10356# | 10363 |
| \$2 | 002066 | 8908 | 8910# |
| \$20 | 002726 | 9133 | 9136# |
| \$200 | 010132 | 10375# | 10398 |
| \$201 | 010170 | 10376 | 10385# |
| \$202 | 010174 | 10385 | 10387# |
| \$203 | 010232 | 10392 | 10397# |
| \$204 | 010232 | 10394 | 10396# |
| \$205 | 010274 | 10414# | 10423 |
| \$206 | 010316 | 10415 | 10418# |
| \$207 | 010326 | 10418 | 10421# |
| \$21 | 002756 | 9145 | 9148# |
| \$210 | 010360 | 10430# | 10431 |
| \$211 | 010374 | 10435# | 10437 |
| \$212 | 010450 | 10451# | 10490 |
| \$213 | 010554 | 10452 | 10480# |
| \$214 | 010510 | 10466# | 10479 |
| \$215 | 010524 | 10467 | 10469# |
| \$216 | 010534 | 10471 | 10473# |
| \$217 | 010554 | 10479# | |
| \$22 | 003016 | 9165 | 9172# |
| \$220 | 010614 | 10483 | 10489# |
| \$221 | 010614 | 10485 | 10488# |
| \$222 | 010632 | 10490# | |
| \$223 | 010644 | 10492 | 10495# |
| \$224 | 010654 | 10496 | 10499# |
| \$225 | 010726 | 10522 | 10527# |
| \$226 | 010732 | 10527 | 10530# |
| \$227 | 011046 | 10549 | 10554# |
| \$23 | 003046 | 9192 | 9194# |
| \$230 | 011052 | 10554 | 10557# |
| \$231 | 011166 | 10576 | 10581# |
| \$232 | 011172 | 10581 | 10584# |
| \$233 | 011306 | 10603 | 10608# |
| \$234 | 011312 | 10608 | 10611# |
| \$235 | 012614 | 10739 | 10748# |
| \$236 | 012616 | 10741 | 10748# |
| \$237 | 012466 | 10712# | 10732 |
| \$24 | 003052 | 9197# | 9259 |
| \$240 | 012570 | 10726 | 10732# |
| \$241 | 012504 | 10714 | 10716# |
| \$242 | 012512 | 10716 | 10718# |
| \$243 | 012530 | 10721 | 10724# |
| \$244 | 012602 | 10738 | 10740# |
| \$245 | 012676 | 10776# | |
| \$246 | 012676 | 10776# | |
| \$247 | 012644 | 10771# | |
| \$25 | 003300 | 9258 | 9259# |
| \$250 | 012640 | 10771# | 10774 |

| | | | | | | | | | | | | | | | | | | | | |
|----------|----------|--------|--------|-------|-------|-------|-------|-------|------|-------|------|------|------|------|--|--|--|--|--|--|
| \$251 | 012670 | 10771 | 10774# | | | | | | | | | | | | | | | | | |
| \$252 | 012656 | 10772# | | | | | | | | | | | | | | | | | | |
| \$253 | 012654 | 10772# | 10773 | | | | | | | | | | | | | | | | | |
| \$254 | 012666 | 10772 | 10773# | | | | | | | | | | | | | | | | | |
| \$255 | 014360 | 11055# | | | | | | | | | | | | | | | | | | |
| \$256 | 014360 | 11055# | | | | | | | | | | | | | | | | | | |
| \$26 | 003126 | 9214 | 9218# | | | | | | | | | | | | | | | | | |
| \$27 | 003166 | 9231 | 9237# | | | | | | | | | | | | | | | | | |
| \$3 | 002130 | 8934# | 8948 | | | | | | | | | | | | | | | | | |
| \$30 | 003222 | 9237 | 9246# | | | | | | | | | | | | | | | | | |
| \$31 | 003222 | 9242 | 9245# | | | | | | | | | | | | | | | | | |
| \$32 | 003260 | 9247 | 9255# | | | | | | | | | | | | | | | | | |
| \$33 | 003256 | 9249 | 9254# | | | | | | | | | | | | | | | | | |
| \$34 | 003266 | 9255 | 9257# | | | | | | | | | | | | | | | | | |
| \$35 | 003346 | 9281# | | | | | | | | | | | | | | | | | | |
| \$36 | 003350 | 9281 | 9283# | | | | | | | | | | | | | | | | | |
| \$37 | 003416 | 9296 | 9299# | | | | | | | | | | | | | | | | | |
| \$4 | 002154 | 8941 | 8944# | | | | | | | | | | | | | | | | | |
| \$40 | 003440 | 9307 | 9310# | | | | | | | | | | | | | | | | | |
| \$40CAT= | ***** U | 11075 | 11078 | | | | | | | | | | | | | | | | | |
| \$41 | 003476 | 9325# | | | | | | | | | | | | | | | | | | |
| \$42 | 003500 | 9325 | 9327# | | | | | | | | | | | | | | | | | |
| \$43 | 003546 | 9339 | 9343# | | | | | | | | | | | | | | | | | |
| \$44 | 003566 | 9350 | 9354# | | | | | | | | | | | | | | | | | |
| \$45 | 003624 | 9368# | | | | | | | | | | | | | | | | | | |
| \$46 | 003626 | 9368 | 9370# | | | | | | | | | | | | | | | | | |
| \$47 | 003674 | 9382 | 9386# | | | | | | | | | | | | | | | | | |
| \$5 | 002314 | 8992 | 8994# | | | | | | | | | | | | | | | | | |
| \$50 | 003712 | 9395 | 9399# | | | | | | | | | | | | | | | | | |
| \$51 | 003754 | 9413 | 9419# | | | | | | | | | | | | | | | | | |
| \$52 | 003750 | 9414 | 9416# | | | | | | | | | | | | | | | | | |
| \$53 | 003752 | 9416 | 9418# | | | | | | | | | | | | | | | | | |
| \$54 | 003756 | 9419 | 9421# | | | | | | | | | | | | | | | | | |
| \$55 | 004024 | 9435 | 9438# | | | | | | | | | | | | | | | | | |
| \$56 | 004062 | 9449 | 9455# | | | | | | | | | | | | | | | | | |
| \$57 | 004102 | 9460 | 9468# | | | | | | | | | | | | | | | | | |
| \$6 | 002334 | 8999 | 9002# | | | | | | | | | | | | | | | | | |
| \$60 | 004124 | 9474 | 9480# | | | | | | | | | | | | | | | | | |
| \$61 | 004172 | 9493 | 9496# | | | | | | | | | | | | | | | | | |
| \$62 | 004206 | 9498 | 9505# | | | | | | | | | | | | | | | | | |
| \$63 | 004220 | 9507 | 9511# | | | | | | | | | | | | | | | | | |
| \$64 | 004404 | 9559 | 9562# | | | | | | | | | | | | | | | | | |
| \$65 | 004432 | 9568 | 9571# | | | | | | | | | | | | | | | | | |
| \$66 | 004550 | 9606# | | | | | | | | | | | | | | | | | | |
| \$67 | 004554 | 9606 | 9608# | | | | | | | | | | | | | | | | | |
| \$7 | 002362 | 9009 | 9012# | | | | | | | | | | | | | | | | | |
| \$70 | 004724 | 9635 | 9638# | | | | | | | | | | | | | | | | | |
| \$71 | 004752 | 9643 | 9646# | | | | | | | | | | | | | | | | | |
| \$72 | 005032 | 9667 | 9670# | | | | | | | | | | | | | | | | | |
| \$73 | 005104 | 9689# | | | | | | | | | | | | | | | | | | |
| \$74 | 005106 | 9689 | 9692# | | | | | | | | | | | | | | | | | |
| \$75 | 005116 | 9695# | | | | | | | | | | | | | | | | | | |
| \$76 | 005114 | 9695# | 9745 | | | | | | | | | | | | | | | | | |
| \$77 | 005322 | 9695 | 9745# | | | | | | | | | | | | | | | | | |
| . | = 017274 | 8834# | 8841# | 8854# | 8855# | 8856# | 8857# | 8866# | 8871 | 8872# | 8903 | 8906 | 8941 | 8948 | | | | | | |
| | | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 | 9122 | 9133 | 9145 | | | | | | |

| | | | | | | | | | | | | |
|--------|--------|--------|--------|--------|--------|-------|-------|--------|--------|--------|--------|--------|
| 9165 | 9192 | 9214 | 9231 | 9242 | 9247 | 9249 | 9281 | 9296 | 9307 | 9325 | 9339 | 9350 |
| 9368 | 9382 | 9395 | 9413 | 9414 | 9435 | 9449 | 9460 | 9474 | 9493 | 9498 | 9507 | 9559 |
| 9568 | 9606 | 9635 | 9643 | 9667 | 9689 | 9701 | 9711 | 9720 | 9727 | 9728 | 9729 | 9739 |
| 9758 | 9792 | 9808 | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 | 9886 | 9918 | 9921 |
| 9935 | 9940 | 9944 | 9947 | 9949 | 9951 | 9956 | 9968 | 9971 | 9974 | 9979 | 9993 | 10009 |
| 10015 | 10034 | 10035 | 10036# | 10038 | 10039# | 10052 | 10081 | 10093 | 10103 | 10116 | 10126 | 10128 |
| 10132 | 10181 | 10311# | 10351 | 10363 | 10376 | 10392 | 10394 | 10398 | 10415 | 10423 | 10431 | 10437 |
| 10452 | 10467 | 10471 | 10479 | 10483 | 10485 | 10490 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 |
| 10631# | 10632# | 10633# | 10634# | 10651# | 10714 | 10721 | 10738 | 10808 | 10847# | 10857# | 10859# | 11043# |
| 11048# | 11050# | 11063 | 11066 | 11069# | 11072# | 11075 | 11078 | 11081# | 11093# | | | |
| 11072 | | | | | | | | | | | | |
| 8856# | | | | | | | | | | | | |

.\$ASTA= ***** U
 .\$X = 001000

| | | | | | | | | | | | | | | | |
|---------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| BEGIN | 7390# | | | | | | | | | | | | | | |
| BGNSRV | 8624# | 9853 | 9875 | 10206 | 10217 | 10227 | 10237 | 10247 | 10257 | 10267 | 10277 | 10518 | 10533 | 10545 | 10560 |
| | 10572 | 10587 | 10599 | 10614 | 10929 | | | | | | | | | | |
| BGNSUB | 8634# | 8935 | 8997 | 9006 | 9016 | 9027 | 9053 | 9062 | 9072 | 9083 | 9110 | 9119 | 9129 | 9140 | 9163 |
| | 9285 | 9303 | 9329 | 9372 | 9425 | 9459 | 9471 | 9483 | 9543 | 9574 | 9614 | 9655 | 9926 | 9986 | |
| BRESET | 8639# | 8904 | 9031 | 9087 | 9144 | 9171 | 9305 | 9674 | 9693 | 9798 | 9962 | 10150 | 10440 | | |
| CALL | 7063# | 9211 | 9241 | 9293 | 9337 | 9380 | 9432 | 9443 | 9490 | 9545 | 9556 | 9565 | 9583 | 9627 | 9632 |
| | 9640 | 9665 | 9700 | 9710 | 9934 | 9990 | 10000 | 10008 | 10114 | 10127 | 10393 | 10484 | 10730 | | |
| CASE | 7329# | | | | | | | | | | | | | | |
| CLRVEC | 8649# | | | | | | | | | | | | | | |
| COMMEN | 1566# | 8724# | | | | | | | | | | | | | |
| DECR | 6996# | | | | | | | | | | | | | | |
| DECRU | 7002# | | | | | | | | | | | | | | |
| DEFAULT | 7372# | | | | | | | | | | | | | | |
| ELSE | 6427# | 8908 | 9237 | 9255 | 9416 | 9419 | 9734 | 9739 | 9794 | 9819 | 9822 | 9825 | 9829 | 9884 | 9888 |
| | 9920 | 9955 | 10037 | 10385 | 10418 | 10527 | 10554 | 10581 | 10608 | 10716 | | | | | |
| END | 7407# | | | | | | | | | | | | | | |
| ENDCOM | 1578# | 8724# | | | | | | | | | | | | | |
| ENDDEC | 7007# | | | | | | | | | | | | | | |
| ENDDO | 6498# | | | | | | | | | | | | | | |
| ENDIF | 6445# | 8910 | 8944 | 8994 | 9002 | 9012 | 9023 | 9035 | 9058 | 9068 | 9079 | 9091 | 9115 | 9125 | 9136 |
| | 9148 | 9172 | 9194 | 9218 | 9245 | 9246 | 9254 | 9257 | 9283 | 9299 | 9310 | 9327 | 9343 | 9354 | 9370 |
| | 9386 | 9399 | 9418 | 9421 | 9438 | 9455 | 9468 | 9480 | 9496 | 9505 | 9511 | 9562 | 9571 | 9608 | 9638 |
| | 9646 | 9670 | 9692 | 9705 | 9715 | 9723 | 9733 | 9738 | 9744 | 9760 | 9796 | 9828 | 9832 | 9833 | 9857 |
| | 9891 | 9892 | 9924 | 9925 | 9938 | 9943 | 9946 | 9954 | 9958 | 9959 | 9960 | 9970 | 9973 | 9978 | 9984 |
| | 9996 | 10012 | 10018 | 10041 | 10042 | 10050 | 10055 | 10119 | 10120 | 10130 | 10131 | 10188 | 10387 | 10396 | 10397 |
| | 10421 | 10469 | 10473 | 10480 | 10488 | 10489 | 10495 | 10499 | 10530 | 10557 | 10584 | 10611 | 10718 | 10724 | 10740 |
| ENDINC | 6983# | 9745 | 10773 | 10774 | | | | | | | | | | | |
| ENDLOO | 6871# | 9259 | 10732 | | | | | | | | | | | | |
| ENDRTN | 7225# | 10748 | 10776 | 11055 | | | | | | | | | | | |
| ENDSEL | 7316# | | | | | | | | | | | | | | |
| ENDSRV | 8660# | 9860 | 9893 | 10214 | 10225 | 10235 | 10245 | 10255 | 10265 | 10275 | 10285 | 10531 | 10543 | 10558 | 10570 |
| | 10585 | 10597 | 10612 | 10624 | 10932 | | | | | | | | | | |
| ENDSUB | 8670# | 8945 | 9003 | 9013 | 9024 | 9036 | 9059 | 9069 | 9080 | 9092 | 9116 | 9126 | 9137 | 9149 | 9174 |
| | 9301 | 9311 | 9344 | 9387 | 9456 | 9469 | 9481 | 9512 | 9572 | 9647 | 9671 | 9985 | 10019 | | |
| ERRDF | 8674# | 8943 | | | | | | | | | | | | | |
| ERRHRD | 8684# | 9001 | 9011 | 9022 | 9034 | 9057 | 9067 | 9078 | 9090 | 9114 | 9124 | 9135 | 9147 | 9169 | 9216 |
| | 9244 | 9252 | 9298 | 9309 | 9341 | 9352 | 9384 | 9397 | 9437 | 9451 | 9463 | 9477 | 9495 | 9500 | 9509 |
| | 9561 | 9570 | 9587 | 9637 | 9645 | 9669 | 9703 | 9712 | 9722 | 9731 | 9736 | 9742 | 9818 | 9821 | 9824 |
| | 9827 | 9831 | 9937 | 9969 | 9972 | 9975 | 9980 | 9995 | 10011 | 10017 | 10118 | 10182 | 10494 | 10498 | |
| ERROR | 8724# | 8943 | 8954 | 8958 | 8962 | 8966 | 9001 | 9011 | 9022 | 9034 | 9057 | 9067 | 9078 | 9090 | 9114 |
| | 9124 | 9135 | 9147 | 9169 | 9216 | 9244 | 9252 | 9298 | 9309 | 9341 | 9352 | 9384 | 9397 | 9437 | 9451 |
| | 9463 | 9477 | 9495 | 9500 | 9509 | 9561 | 9570 | 9587 | 9637 | 9645 | 9669 | 9703 | 9712 | 9722 | 9731 |
| | 9736 | 9742 | 9818 | 9821 | 9824 | 9827 | 9831 | 9937 | 9965 | 9969 | 9972 | 9975 | 9980 | 9995 | 10011 |
| | 10017 | 10118 | 10182 | 10494 | 10498 | | | | | | | | | | |
| ESCAPE | 1698# | 8724# | | | | | | | | | | | | | |
| EXIF | 6829# | 9258 | | | | | | | | | | | | | |
| EXIFB | 6851# | 10726 | | | | | | | | | | | | | |
| EXIT | 8694# | 8993 | 9193 | 9260 | 9282 | 9326 | 9369 | 9417 | 9420 | 9454 | 9467 | 9479 | 9504 | 9513 | 9691 |
| | 9704 | 9714 | 9732 | 9737 | 9743 | 9834 | 10020 | 10190 | | | | | | | |
| GETPRI | 1312# | 8724# | | | | | | | | | | | | | |
| GETSWR | 1771# | 8724# | 8872# | | | | | | | | | | | | |
| IF | 6332# | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 | 9122 | 9133 | 9145 |
| | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 | 9395 | 9413 | 9414 | 9449 | 9460 |
| | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 | 9727 | 9728 | 9729 | 9758 | 9792 | 9814 | 9815 |
| | 9816 | 9854 | 9879 | 9886 | 9918 | 9921 | 9944 | 9947 | 9949 | 9951 | 9956 | 9968 | 9971 | 9974 | 9979 |

| | | | | | | | | | | | | | | | |
|---------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| SPACE | 8724# | | | | | | | | | | | | | | |
| STARS | 1485# | 8724# | 8739 | 8741 | 8759 | 8761 | 8781 | 8783 | 8802 | 8804 | 8823 | 8833 | 8854 | 8856 | 8857 |
| | 8923 | 8977 | 8979 | 8986 | 9044 | 9104 | 9157 | 9186 | 9275 | 9319 | 9362 | 9407 | 9528 | 9601 | 9684 |
| | 9753 | 9846 | 9848 | 9868 | 9870 | 9910 | 10029 | 10070 | 10198 | 10200 | 10338 | 10510 | 10512 | 10678 | 10697 |
| | 10756 | 10763 | 10919 | 10924 | 10945 | 10949 | 11063 | 11066 | 11069 | 11072 | 11075 | 11081 | 11084 | 11087 | |
| STRUCT | 5720# | 8611# | 8612 | | | | | | | | | | | | |
| SWRSU | 1453# | 8724# | 8871# | | | | | | | | | | | | |
| TRMTRP | 11087# | | | | | | | | | | | | | | |
| TYPBIN | 2088# | 8724# | | | | | | | | | | | | | |
| TYPDEC | 2058# | 8724# | 10048 | 11063 | | | | | | | | | | | |
| TYPNAM | 1826# | 8724# | 8872 | | | | | | | | | | | | |
| TYPNUM | 2025# | 8724# | | | | | | | | | | | | | |
| TYPOCS | 1978# | 8724# | 11047 | | | | | | | | | | | | |
| TYPOCT | 1941# | 8724# | 10044 | 10046 | 11044 | 11049 | 11051 | 11053 | 11069 | | | | | | |
| TYPTXT | 1894# | 8724# | 10036 | 10039 | 10043 | 10045 | 10047 | 10807 | 10821 | 10847 | 10857 | 10859 | 11043 | 11045 | 11048 |
| | 11050 | 11052 | | | | | | | | | | | | | |
| UNTIL | 6650# | 8948 | 9808 | 10081 | 10093 | 10132 | 10351 | 10363 | 10398 | 10423 | 10431 | 10437 | 10479 | 10490 | |
| UNTILB | 6671# | | | | | | | | | | | | | | |
| WAITMS | 8714# | 9443 | 9565 | 9640 | 10000 | 10730 | | | | | | | | | |
| WHILE | 6487# | | | | | | | | | | | | | | |
| WHILEB | 6492# | | | | | | | | | | | | | | |
| \$ADDON | 5805# | 8906 | 8908 | 8934 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 |
| | 9122 | 9133 | 9145 | 9165 | 9192 | 9197 | 9214 | 9231 | 9237 | 9242 | 9247 | 9249 | 9255 | 9259 | 9281 |
| | 9296 | 9307 | 9325 | 9339 | 9350 | 9368 | 9382 | 9395 | 9413 | 9414 | 9416 | 9419 | 9435 | 9449 | 9460 |
| | 9474 | 9493 | 9498 | 9507 | 9559 | 9568 | 9606 | 9635 | 9643 | 9667 | 9689 | 9695 | 9701 | 9711 | 9720 |
| | 9727 | 9728 | 9729 | 9734 | 9739 | 9758 | 9792 | 9794 | 9807 | 9808 | 9814 | 9815 | 9816 | 9819 | 9822 |
| | 9825 | 9829 | 9854 | 9879 | 9884 | 9886 | 9888 | 9918 | 9920 | 9921 | 9935 | 9940 | 9944 | 9947 | 9949 |
| | 9951 | 9955 | 9956 | 9968 | 9971 | 9974 | 9979 | 9993 | 10009 | 10015 | 10034 | 10035 | 10037 | 10038 | 10052 |
| | 10079 | 10089 | 10102 | 10103 | 10116 | 10126 | 10128 | 10181 | 10349 | 10356 | 10375 | 10376 | 10385 | 10392 | 10394 |
| | 10414 | 10415 | 10418 | 10430 | 10435 | 10451 | 10452 | 10466 | 10467 | 10471 | 10479 | 10483 | 10485 | 10490 | 10492 |
| | 10496 | 10522 | 10527 | 10549 | 10554 | 10576 | 10581 | 10603 | 10608 | 10704 | 10712 | 10714 | 10716 | 10721 | 10732 |
| | 10738 | 10768 | 10771 | 10772 | 11042 | | | | | | | | | | |
| \$AND | 6092# | 9739 | 9854 | | | | | | | | | | | | |
| \$BRANC | 5931# | 8906 | 8908 | 8941 | 8948 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 |
| | 9122 | 9133 | 9145 | 9165 | 9192 | 9214 | 9231 | 9237 | 9242 | 9247 | 9249 | 9255 | 9258 | 9259 | 9281 |
| | 9296 | 9307 | 9325 | 9339 | 9350 | 9368 | 9382 | 9395 | 9413 | 9414 | 9416 | 9419 | 9435 | 9449 | 9460 |
| | 9474 | 9493 | 9498 | 9507 | 9559 | 9568 | 9606 | 9635 | 9643 | 9667 | 9689 | 9695 | 9701 | 9711 | 9720 |
| | 9727 | 9728 | 9729 | 9734 | 9739 | 9745 | 9758 | 9792 | 9794 | 9808 | 9814 | 9815 | 9816 | 9819 | 9822 |
| | 9825 | 9829 | 9854 | 9879 | 9884 | 9886 | 9888 | 9918 | 9920 | 9921 | 9935 | 9940 | 9944 | 9947 | 9949 |
| | 9951 | 9955 | 9956 | 9968 | 9971 | 9974 | 9979 | 9993 | 10009 | 10015 | 10034 | 10035 | 10037 | 10038 | 10052 |
| | 10081 | 10093 | 10103 | 10116 | 10126 | 10128 | 10132 | 10181 | 10351 | 10363 | 10376 | 10385 | 10392 | 10394 | 10398 |
| | 10415 | 10418 | 10423 | 10431 | 10437 | 10452 | 10467 | 10471 | 10479 | 10483 | 10485 | 10490 | 10492 | 10496 | 10522 |
| | 10527 | 10549 | 10554 | 10576 | 10581 | 10603 | 10608 | 10714 | 10716 | 10721 | 10726 | 10732 | 10738 | 10739 | 10741 |
| | 10771 | 10772 | 10773 | 10774 | | | | | | | | | | | |
| \$BRCOD | 6230# | 9258 | 9281 | 9325 | 9368 | 9606 | 9689 | 9695 | 9758 | 9808 | 9879 | 9918 | 10479 | 10490 | 10726 |
| | 10771 | 10772 | | | | | | | | | | | | | |
| \$CALL | 7039# | 9211 | 9241 | 9293 | 9337 | 9380 | 9432 | 9443 | 9490 | 9545 | 9556 | 9565 | 9583 | 9627 | 9632 |
| | 9640 | 9665 | 9700 | 9710 | 9934 | 9990 | 10000 | 10008 | 10114 | 10127 | 10393 | 10484 | 10730 | | |
| \$CHECK | 6288# | 8906 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 | 9122 | 9133 |
| | 9145 | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 | 9395 | 9413 | 9414 | 9449 |
| | 9460 | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 | 9727 | 9728 | 9729 | 9739 | 9758 | 9792 |
| | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 | 9886 | 9918 | 9921 | 9940 | 9944 | 9947 | 9949 | 9951 |
| | 9956 | 9968 | 9971 | 9974 | 9979 | 10015 | 10034 | 10035 | 10038 | 10052 | 10103 | 10126 | 10128 | 10181 | 10376 |
| | 10392 | 10394 | 10415 | 10452 | 10467 | 10471 | 10483 | 10485 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 | 10714 |
| | 10721 | 10738 | | | | | | | | | | | | | |
| \$CHK1 | 7725# | 8892 | 8894 | 8896 | 8900 | 8909 | 8928 | 8933 | 8935 | 8937 | 8997 | 9006 | 9016 | 9027 | 9053 |

| | | | | | | | | | | | | | | | |
|---------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| \$GENTA | 5900# | 8908 | 8910 | 8934 | 8944 | 8994 | 9002 | 9012 | 9023 | 9035 | 9058 | 9068 | 9079 | 9091 | 9115 |
| | 9125 | 9136 | 9148 | 9172 | 9194 | 9197 | 9218 | 9237 | 9245 | 9246 | 9254 | 9255 | 9257 | 9259 | 9281 |
| | 9283 | 9299 | 9310 | 9325 | 9327 | 9343 | 9354 | 9368 | 9370 | 9386 | 9399 | 9416 | 9418 | 9419 | 9421 |
| | 9438 | 9455 | 9468 | 9480 | 9496 | 9505 | 9511 | 9562 | 9571 | 9606 | 9608 | 9638 | 9646 | 9670 | 9689 |
| | 9692 | 9695 | 9705 | 9715 | 9723 | 9733 | 9734 | 9738 | 9739 | 9744 | 9745 | 9758 | 9760 | 9794 | 9796 |
| | 9807 | 9808 | 9819 | 9822 | 9825 | 9828 | 9829 | 9832 | 9833 | 9857 | 9879 | 9884 | 9888 | 9891 | 9892 |
| | 9918 | 9920 | 9924 | 9925 | 9938 | 9943 | 9946 | 9954 | 9955 | 9958 | 9959 | 9960 | 9970 | 9973 | 9978 |
| | 9984 | 9996 | 10012 | 10018 | 10037 | 10041 | 10042 | 10050 | 10055 | 10079 | 10089 | 10102 | 10119 | 10120 | 10130 |
| | 10131 | 10188 | 10349 | 10356 | 10375 | 10385 | 10387 | 10396 | 10397 | 10414 | 10418 | 10421 | 10430 | 10435 | 10451 |
| | 10466 | 10469 | 10473 | 10479 | 10480 | 10488 | 10489 | 10490 | 10495 | 10499 | 10527 | 10530 | 10554 | 10557 | 10581 |
| | 10584 | 10608 | 10611 | 10712 | 10716 | 10718 | 10724 | 10732 | 10740 | 10748 | 10771 | 10772 | 10773 | 10774 | 10776 |
| | 11055 | | | | | | | | | | | | | | |
| \$IF | 6306# | 8906 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 | 9122 | 9133 |
| | 9145 | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 | 9395 | 9413 | 9414 | 9449 |
| | 9460 | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 | 9727 | 9728 | 9729 | 9739 | 9758 | 9792 |
| | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 | 9886 | 9918 | 9921 | 9940 | 9944 | 9947 | 9949 | 9951 |
| | 9956 | 9968 | 9971 | 9974 | 9979 | 10015 | 10034 | 10035 | 10038 | 10052 | 10103 | 10126 | 10128 | 10181 | 10376 |
| | 10392 | 10394 | 10415 | 10452 | 10467 | 10471 | 10483 | 10485 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 | 10714 |
| | 10721 | 10738 | | | | | | | | | | | | | |
| \$IFCOD | 6178# | 8906 | 8941 | 8948 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 | 9122 |
| | 9133 | 9145 | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 | 9395 | 9413 | 9414 |
| | 9449 | 9460 | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 | 9727 | 9728 | 9729 | 9739 | 9758 |
| | 9792 | 9808 | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 | 9886 | 9918 | 9921 | 9940 | 9944 | 9947 |
| | 9949 | 9951 | 9956 | 9968 | 9971 | 9974 | 9979 | 10015 | 10034 | 10035 | 10038 | 10052 | 10081 | 10093 | 10103 |
| | 10126 | 10128 | 10132 | 10181 | 10351 | 10363 | 10376 | 10392 | 10394 | 10398 | 10415 | 10423 | 10431 | 10437 | 10452 |
| | 10467 | 10471 | 10479 | 10483 | 10485 | 10490 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 | 10714 | 10721 | 10738 |
| \$IFCON | 6345# | 9214 | 9242 | 9296 | 9339 | 9382 | 9435 | 9493 | 9559 | 9635 | 9667 | 9701 | 9711 | 9935 | 9993 |
| | 10009 | 10116 | | | | | | | | | | | | | |
| \$IFOPR | 5944# | 8906 | 8941 | 8948 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 | 9122 |
| | 9133 | 9145 | 9165 | 9192 | 9214 | 9231 | 9242 | 9247 | 9249 | 9281 | 9296 | 9307 | 9325 | 9339 | 9350 |
| | 9368 | 9382 | 9395 | 9413 | 9414 | 9435 | 9449 | 9460 | 9474 | 9493 | 9498 | 9507 | 9559 | 9568 | 9606 |
| | 9635 | 9643 | 9667 | 9689 | 9701 | 9711 | 9720 | 9727 | 9728 | 9729 | 9739 | 9758 | 9792 | 9808 | 9814 |
| | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 | 9886 | 9918 | 9921 | 9935 | 9940 | 9944 | 9947 | 9949 | 9951 |
| | 9956 | 9968 | 9971 | 9974 | 9979 | 9993 | 10009 | 10015 | 10034 | 10035 | 10038 | 10052 | 10081 | 10093 | 10103 |
| | 10116 | 10126 | 10128 | 10132 | 10181 | 10351 | 10363 | 10376 | 10392 | 10394 | 10398 | 10415 | 10423 | 10431 | 10437 |
| | 10452 | 10467 | 10471 | 10479 | 10483 | 10485 | 10490 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 | 10714 | 10721 |
| | 10738 | | | | | | | | | | | | | | |
| \$LET | 7909# | 8892 | 8894 | 8896 | 8897 | 8898 | 8899 | 8900 | 8905 | 8909 | 8928 | 8933 | 8935 | 8937 | 8946 |
| | 8947 | 8997 | 9006 | 9007 | 9016 | 9018 | 9027 | 9029 | 9053 | 9062 | 9063 | 9072 | 9074 | 9083 | 9085 |
| | 9110 | 9119 | 9120 | 9129 | 9131 | 9140 | 9142 | 9163 | 9196 | 9199 | 9200 | 9207 | 9227 | 9234 | 9248 |
| | 9253 | 9256 | 9285 | 9288 | 9303 | 9329 | 9333 | 9349 | 9372 | 9376 | 9393 | 9425 | 9430 | 9441 | 9446 |
| | 9459 | 9471 | 9483 | 9489 | 9534 | 9537 | 9539 | 9543 | 9548 | 9554 | 9574 | 9576 | 9578 | 9581 | 9612 |
| | 9614 | 9615 | 9617 | 9623 | 9629 | 9655 | 9658 | 9660 | 9662 | 9708 | 9718 | 9780 | 9782 | 9783 | 9786 |
| | 9787 | 9790 | 9791 | 9793 | 9795 | 9801 | 9803 | 9812 | 9813 | 9856 | 9859 | 9877 | 9881 | 9882 | 9883 |
| | 9885 | 9887 | 9890 | 9926 | 9928 | 9930 | 9932 | 9939 | 9942 | 9945 | 9953 | 9957 | 9986 | 9988 | 9999 |
| | 10004 | 10006 | 10040 | 10053 | 10054 | 10078 | 10080 | 10087 | 10088 | 10090 | 10091 | 10092 | 10099 | 10100 | 10101 |
| | 10105 | 10106 | 10108 | 10110 | 10112 | 10122 | 10124 | 10125 | 10129 | 10144 | 10152 | 10154 | 10155 | 10157 | 10208 |
| | 10210 | 10211 | 10213 | 10219 | 10221 | 10222 | 10224 | 10229 | 10231 | 10232 | 10234 | 10239 | 10241 | 10242 | 10244 |
| | 10249 | 10251 | 10252 | 10254 | 10259 | 10261 | 10262 | 10264 | 10269 | 10271 | 10272 | 10274 | 10279 | 10281 | 10282 |
| | 10284 | 10344 | 10348 | 10350 | 10354 | 10355 | 10357 | 10358 | 10359 | 10360 | 10361 | 10362 | 10369 | 10371 | 10372 |
| | 10374 | 10378 | 10380 | 10381 | 10383 | 10384 | 10386 | 10388 | 10391 | 10395 | 10401 | 10402 | 10403 | 10404 | 10409 |
| | 10411 | 10413 | 10416 | 10417 | 10419 | 10420 | 10422 | 10426 | 10434 | 10436 | 10444 | 10445 | 10446 | 10447 | 10449 |
| | 10450 | 10453 | 10455 | 10457 | 10459 | 10461 | 10462 | 10464 | 10468 | 10472 | 10475 | 10477 | 10481 | 10482 | 10487 |
| | 10519 | 10520 | 10521 | 10524 | 10526 | 10529 | 10534 | 10535 | 10536 | 10537 | 10539 | 10542 | 10546 | 10547 | 10548 |
| | 10551 | 10553 | 10556 | 10561 | 10562 | 10563 | 10564 | 10566 | 10569 | 10573 | 10574 | 10575 | 10578 | 10580 | 10583 |
| | 10588 | 10589 | 10590 | 10591 | 10593 | 10596 | 10600 | 10601 | 10602 | 10605 | 10607 | 10610 | 10615 | 10616 | 10617 |

| | | | | | | | | | | | | | | | |
|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| | 10618 | 10620 | 10623 | 10705 | 10706 | 10707 | 10715 | 10717 | 10723 | 10731 | 10770 | 10931 | | | |
| SLPCNT | 6927# | 9695 | 10771 | 10772 | | | | | | | | | | | |
| SOPADD | 7566# | 8897 | 8898 | 8899 | 8905 | 8946 | 8947 | 9248 | 9539 | 9695 | 9883 | 9885 | 10040 | 10054 | 10092 |
| | 10106 | 10122 | 10125 | 10129 | 10211 | 10222 | 10232 | 10242 | 10252 | 10262 | 10272 | 10282 | 10359 | 10362 | 10378 |
| | 10381 | 10384 | 10386 | 10391 | 10395 | 10422 | 10462 | 10472 | 10475 | 10477 | 10481 | 10487 | 10520 | 10524 | 10535 |
| | 10537 | 10547 | 10551 | 10562 | 10564 | 10574 | 10578 | 10589 | 10591 | 10601 | 10605 | 10616 | 10618 | 10771 | 10772 |
| 10931 | | | | | | | | | | | | | | | |
| SOPAND | 7608# | | | | | | | | | | | | | | |
| SOPCD1 | 7649# | 8897 | 8898 | 8899 | 8905 | 8946 | 8947 | 9007 | 9018 | 9029 | 9063 | 9074 | 9085 | 9120 | 9131 |
| | 9142 | 9248 | 9349 | 9539 | 9548 | 9554 | 9576 | 9617 | 9629 | 9658 | 9695 | 9801 | 9803 | 9812 | 9813 |
| | 9859 | 9882 | 9883 | 9885 | 9887 | 9890 | 9930 | 9942 | 9945 | 9953 | 9957 | 9988 | 9999 | 10040 | 10054 |
| | 10092 | 10105 | 10106 | 10108 | 10112 | 10122 | 10124 | 10125 | 10129 | 10211 | 10213 | 10222 | 10224 | 10232 | 10234 |
| | 10242 | 10244 | 10252 | 10254 | 10262 | 10264 | 10272 | 10274 | 10282 | 10284 | 10359 | 10362 | 10378 | 10380 | 10381 |
| | 10383 | 10384 | 10386 | 10388 | 10391 | 10395 | 10422 | 10436 | 10455 | 10459 | 10462 | 10468 | 10472 | 10475 | 10477 |
| | 10481 | 10482 | 10487 | 10520 | 10524 | 10526 | 10529 | 10535 | 10537 | 10539 | 10542 | 10547 | 10551 | 10553 | 10556 |
| | 10562 | 10564 | 10566 | 10569 | 10574 | 10578 | 10580 | 10583 | 10589 | 10591 | 10593 | 10596 | 10601 | 10605 | 10607 |
| | 10610 | 10616 | 10618 | 10620 | 10623 | 10731 | 10771 | 10772 | 10931 | | | | | | |
| SOPCD2 | 7524# | 10124 | 10388 | 10455 | 10459 | 10482 | | | | | | | | | |
| SOPCOD | 7710# | 8897 | 8898 | 8899 | 8905 | 8946 | 8947 | 9007 | 9018 | 9029 | 9063 | 9074 | 9085 | 9120 | 9131 |
| | 9142 | 9248 | 9349 | 9539 | 9548 | 9554 | 9576 | 9617 | 9629 | 9658 | 9695 | 9801 | 9803 | 9812 | 9813 |
| | 9859 | 9882 | 9883 | 9885 | 9887 | 9890 | 9930 | 9942 | 9945 | 9953 | 9957 | 9988 | 9999 | 10040 | 10054 |
| | 10092 | 10105 | 10106 | 10108 | 10112 | 10122 | 10124 | 10125 | 10129 | 10211 | 10213 | 10222 | 10224 | 10232 | 10234 |
| | 10242 | 10244 | 10252 | 10254 | 10262 | 10264 | 10272 | 10274 | 10282 | 10284 | 10359 | 10362 | 10378 | 10380 | 10381 |
| | 10383 | 10384 | 10386 | 10388 | 10391 | 10395 | 10422 | 10436 | 10455 | 10459 | 10462 | 10468 | 10472 | 10475 | 10477 |
| | 10481 | 10482 | 10487 | 10520 | 10524 | 10526 | 10529 | 10535 | 10537 | 10539 | 10542 | 10547 | 10551 | 10553 | 10556 |
| | 10562 | 10564 | 10566 | 10569 | 10574 | 10578 | 10580 | 10583 | 10589 | 10591 | 10593 | 10596 | 10601 | 10605 | 10607 |
| | 10610 | 10616 | 10618 | 10620 | 10623 | 10731 | 10771 | 10772 | 10931 | | | | | | |
| SOPCOM | 7461# | | | | | | | | | | | | | | |
| SOPDEF | 5909# | 8892 | 8894 | 8896 | 8897 | 8898 | 8899 | 8900 | 8905 | 8906 | 8908 | 8909 | 8928 | 8933 | 8935 |
| | 8937 | 8941 | 8946 | 8947 | 8948 | 8992 | 8997 | 8999 | 9006 | 9007 | 9009 | 9016 | 9018 | 9020 | 9027 |
| | 9029 | 9032 | 9053 | 9055 | 9062 | 9063 | 9065 | 9072 | 9074 | 9076 | 9083 | 9085 | 9088 | 9110 | 9112 |
| | 9119 | 9120 | 9122 | 9129 | 9131 | 9133 | 9140 | 9142 | 9145 | 9163 | 9165 | 9192 | 9196 | 9199 | 9200 |
| | 9207 | 9211 | 9214 | 9227 | 9231 | 9234 | 9237 | 9241 | 9242 | 9247 | 9248 | 9249 | 9253 | 9255 | 9256 |
| | 9258 | 9259 | 9281 | 9285 | 9288 | 9293 | 9296 | 9303 | 9307 | 9325 | 9329 | 9333 | 9337 | 9339 | 9349 |
| | 9350 | 9368 | 9372 | 9376 | 9380 | 9382 | 9393 | 9395 | 9413 | 9414 | 9416 | 9419 | 9425 | 9430 | 9432 |
| | 9435 | 9441 | 9443 | 9446 | 9449 | 9459 | 9460 | 9471 | 9474 | 9483 | 9489 | 9490 | 9493 | 9498 | 9507 |
| | 9534 | 9537 | 9539 | 9543 | 9545 | 9548 | 9554 | 9556 | 9559 | 9565 | 9568 | 9574 | 9576 | 9578 | 9581 |
| | 9583 | 9606 | 9612 | 9614 | 9615 | 9617 | 9623 | 9627 | 9629 | 9632 | 9635 | 9640 | 9643 | 9655 | 9658 |
| | 9660 | 9662 | 9665 | 9667 | 9689 | 9695 | 9700 | 9701 | 9708 | 9710 | 9711 | 9718 | 9720 | 9727 | 9728 |
| | 9729 | 9734 | 9739 | 9745 | 9758 | 9780 | 9782 | 9783 | 9786 | 9787 | 9790 | 9791 | 9792 | 9793 | 9794 |
| | 9795 | 9801 | 9803 | 9808 | 9812 | 9813 | 9814 | 9815 | 9816 | 9819 | 9822 | 9825 | 9829 | 9854 | 9856 |
| | 9859 | 9877 | 9879 | 9881 | 9882 | 9883 | 9884 | 9885 | 9886 | 9887 | 9888 | 9890 | 9918 | 9920 | 9921 |
| | 9926 | 9928 | 9930 | 9932 | 9934 | 9935 | 9939 | 9940 | 9942 | 9944 | 9945 | 9947 | 9949 | 9951 | 9953 |
| | 9955 | 9956 | 9957 | 9968 | 9971 | 9974 | 9979 | 9986 | 9988 | 9990 | 9993 | 9999 | 10000 | 10004 | 10006 |
| | 10008 | 10009 | 10015 | 10034 | 10035 | 10037 | 10038 | 10040 | 10052 | 10053 | 10054 | 10078 | 10080 | 10081 | 10087 |
| | 10088 | 10090 | 10091 | 10092 | 10093 | 10099 | 10100 | 10101 | 10103 | 10105 | 10106 | 10108 | 10110 | 10112 | 10114 |
| | 10116 | 10122 | 10124 | 10125 | 10126 | 10127 | 10128 | 10129 | 10132 | 10144 | 10152 | 10154 | 10155 | 10157 | 10181 |
| | 10208 | 10210 | 10211 | 10213 | 10219 | 10221 | 10222 | 10224 | 10229 | 10231 | 10232 | 10234 | 10239 | 10241 | 10242 |
| | 10244 | 10249 | 10251 | 10252 | 10254 | 10259 | 10261 | 10262 | 10264 | 10269 | 10271 | 10272 | 10274 | 10279 | 10281 |
| | 10282 | 10284 | 10344 | 10348 | 10350 | 10351 | 10354 | 10355 | 10357 | 10358 | 10359 | 10360 | 10361 | 10362 | 10363 |
| | 10369 | 10371 | 10372 | 10374 | 10376 | 10378 | 10380 | 10381 | 10383 | 10384 | 10385 | 10386 | 10388 | 10391 | 10392 |
| | 10393 | 10394 | 10395 | 10398 | 10401 | 10402 | 10403 | 10404 | 10409 | 10411 | 10413 | 10415 | 10416 | 10417 | 10418 |
| | 10419 | 10420 | 10422 | 10423 | 10426 | 10431 | 10434 | 10436 | 10437 | 10444 | 10445 | 10446 | 10447 | 10449 | 10450 |
| | 10452 | 10453 | 10455 | 10457 | 10459 | 10461 | 10462 | 10464 | 10467 | 10468 | 10471 | 10472 | 10475 | 10477 | 10479 |
| | 10481 | 10482 | 10483 | 10484 | 10485 | 10487 | 10490 | 10492 | 10496 | 10519 | 10520 | 10521 | 10522 | 10524 | 10526 |
| | 10527 | 10529 | 10534 | 10535 | 10536 | 10537 | 10539 | 10542 | 10546 | 10547 | 10548 | 10549 | 10551 | 10553 | 10554 |

| | | | | | | | | | | | | | | | |
|---------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| | 10556 | 10561 | 10562 | 10563 | 10564 | 10566 | 10569 | 10573 | 10574 | 10575 | 10576 | 10578 | 10580 | 10581 | 10583 |
| | 10588 | 10589 | 10590 | 10591 | 10593 | 10596 | 10600 | 10601 | 10602 | 10603 | 10605 | 10607 | 10608 | 10610 | 10615 |
| | 10616 | 10617 | 10618 | 10620 | 10623 | 10705 | 10706 | 10707 | 10714 | 10715 | 10716 | 10717 | 10721 | 10723 | 10726 |
| | 10730 | 10731 | 10732 | 10738 | 10739 | 10741 | 10748 | 10770 | 10771 | 10772 | 10773 | 10774 | 10776 | 10931 | 11055 |
| \$OPEQU | 7642# | | | | | | | | | | | | | | |
| \$OPNAN | 7627# | | | | | | | | | | | | | | |
| \$OPNEG | 7473# | | | | | | | | | | | | | | |
| \$OPNOR | 7621# | | | | | | | | | | | | | | |
| \$OPNOT | 7616# | 9018 | 9074 | 9131 | 9548 | 9576 | 9617 | 9658 | 9812 | 9813 | 9859 | 9882 | 9887 | 9999 | 10213 |
| | 10224 | 10234 | 10244 | 10254 | 10264 | 10274 | 10284 | 10542 | 10569 | 10596 | 10623 | | | | |
| \$OPOR | 7603# | 9007 | 9029 | 9063 | 9085 | 9120 | 9142 | 9349 | 9554 | 9629 | 9801 | 9803 | 9890 | 9930 | 9942 |
| | 9945 | 9953 | 9957 | 9988 | 10105 | 10108 | 10380 | 10383 | 10468 | 10526 | 10553 | 10580 | 10607 | | |
| \$OPROT | 7493# | | | | | | | | | | | | | | |
| \$OPRO | 7752# | 8892 | 8894 | 8896 | 8900 | 8909 | 8928 | 8933 | 8935 | 8937 | 8997 | 9006 | 9016 | 9027 | 9053 |
| | 9062 | 9072 | 9083 | 9110 | 9119 | 9129 | 9140 | 9163 | 9196 | 9199 | 9200 | 9207 | 9227 | 9234 | 9253 |
| | 9256 | 9285 | 9288 | 9303 | 9329 | 9333 | 9372 | 9376 | 9393 | 9425 | 9430 | 9441 | 9446 | 9459 | 9471 |
| | 9483 | 9489 | 9534 | 9537 | 9543 | 9574 | 9578 | 9581 | 9612 | 9614 | 9615 | 9623 | 9655 | 9660 | 9662 |
| | 9695 | 9708 | 9718 | 9780 | 9782 | 9783 | 9786 | 9787 | 9790 | 9791 | 9793 | 9795 | 9856 | 9877 | 9881 |
| | 9926 | 9928 | 9932 | 9939 | 9986 | 10004 | 10006 | 10053 | 10078 | 10080 | 10087 | 10088 | 10090 | 10091 | 10099 |
| | 10100 | 10101 | 10110 | 10144 | 10152 | 10154 | 10155 | 10157 | 10208 | 10210 | 10219 | 10221 | 10229 | 10231 | 10239 |
| | 10241 | 10249 | 10251 | 10259 | 10261 | 10269 | 10271 | 10279 | 10281 | 10344 | 10348 | 10350 | 10354 | 10355 | 10357 |
| | 10358 | 10360 | 10361 | 10369 | 10371 | 10372 | 10374 | 10401 | 10402 | 10403 | 10404 | 10409 | 10411 | 10413 | 10416 |
| | 10417 | 10419 | 10420 | 10426 | 10434 | 10444 | 10445 | 10446 | 10447 | 10449 | 10450 | 10453 | 10457 | 10461 | 10464 |
| | 10519 | 10521 | 10534 | 10536 | 10546 | 10548 | 10561 | 10563 | 10573 | 10575 | 10588 | 10590 | 10600 | 10602 | 10615 |
| | 10617 | 10705 | 10706 | 10707 | 10715 | 10717 | 10723 | 10770 | 10771 | 10772 | | | | | |
| \$OPR1 | 7799# | 9695 | 10771 | 10772 | | | | | | | | | | | |
| \$OPR2 | 7868# | 8897 | 8898 | 8899 | 8905 | 8946 | 8947 | 9007 | 9018 | 9029 | 9063 | 9074 | 9085 | 9120 | 9131 |
| | 9142 | 9248 | 9349 | 9539 | 9548 | 9554 | 9576 | 9617 | 9629 | 9658 | 9801 | 9803 | 9812 | 9813 | 9859 |
| | 9882 | 9883 | 9885 | 9887 | 9890 | 9930 | 9942 | 9945 | 9953 | 9957 | 9988 | 9999 | 10040 | 10054 | 10092 |
| | 10105 | 10106 | 10108 | 10112 | 10122 | 10124 | 10125 | 10129 | 10211 | 10213 | 10222 | 10224 | 10232 | 10234 | 10242 |
| | 10244 | 10252 | 10254 | 10262 | 10264 | 10272 | 10274 | 10282 | 10284 | 10359 | 10362 | 10378 | 10380 | 10381 | 10383 |
| | 10384 | 10386 | 10388 | 10391 | 10395 | 10422 | 10436 | 10455 | 10459 | 10462 | 10468 | 10472 | 10475 | 10477 | 10481 |
| | 10482 | 10487 | 10520 | 10524 | 10526 | 10529 | 10535 | 10537 | 10539 | 10542 | 10547 | 10551 | 10553 | 10556 | 10562 |
| | 10564 | 10566 | 10569 | 10574 | 10578 | 10580 | 10583 | 10589 | 10591 | 10593 | 10596 | 10601 | 10605 | 10607 | 10610 |
| | 10616 | 10618 | 10620 | 10623 | 10731 | 10931 | | | | | | | | | |
| \$OPSHF | 7510# | 10124 | 10388 | 10455 | 10459 | 10482 | | | | | | | | | |
| \$OPSUB | 7586# | 10112 | 10436 | 10529 | 10539 | 10556 | 10566 | 10583 | 10593 | 10610 | 10620 | 10731 | | | |
| \$OPSWB | 7485# | | | | | | | | | | | | | | |
| \$OPXOR | 7633# | | | | | | | | | | | | | | |
| \$OR | 6065# | 9281 | 9325 | 9368 | 9606 | 9689 | 9758 | 9879 | 9918 | | | | | | |
| \$PUT | 7022# | 9211 | 9241 | 9293 | 9337 | 9380 | 9432 | 9443 | 9490 | 9545 | 9556 | 9565 | 9583 | 9627 | 9632 |
| | 9640 | 9665 | 9700 | 9710 | 9934 | 9990 | 10000 | 10008 | 10114 | 10730 | | | | | |
| \$STRUC | 5765# | | | | | | | | | | | | | | |
| \$SUBON | 5811# | 8908 | 8910 | 8944 | 8948 | 8994 | 9002 | 9012 | 9023 | 9035 | 9058 | 9068 | 9079 | 9091 | 9115 |
| | 9125 | 9136 | 9148 | 9172 | 9194 | 9218 | 9237 | 9245 | 9246 | 9254 | 9255 | 9257 | 9259 | 9283 | 9299 |
| | 9310 | 9327 | 9343 | 9354 | 9370 | 9386 | 9399 | 9416 | 9418 | 9419 | 9421 | 9438 | 9455 | 9468 | 9480 |
| | 9496 | 9505 | 9511 | 9562 | 9571 | 9608 | 9638 | 9646 | 9670 | 9692 | 9695 | 9705 | 9715 | 9723 | 9733 |
| | 9734 | 9738 | 9739 | 9744 | 9745 | 9760 | 9794 | 9796 | 9808 | 9819 | 9822 | 9825 | 9828 | 9829 | 9832 |
| | 9833 | 9857 | 9884 | 9888 | 9891 | 9892 | 9920 | 9924 | 9925 | 9938 | 9943 | 9946 | 9954 | 9955 | 9958 |
| | 9959 | 9960 | 9970 | 9973 | 9978 | 9984 | 9996 | 10012 | 10018 | 10037 | 10041 | 10042 | 10050 | 10055 | 10081 |
| | 10093 | 10119 | 10120 | 10130 | 10131 | 10132 | 10188 | 10351 | 10363 | 10385 | 10387 | 10396 | 10397 | 10398 | 10418 |
| | 10421 | 10423 | 10431 | 10437 | 10469 | 10473 | 10479 | 10480 | 10488 | 10489 | 10490 | 10495 | 10499 | 10527 | 10530 |
| | 10554 | 10557 | 10581 | 10584 | 10608 | 10611 | 10716 | 10718 | 10724 | 10732 | 10740 | 10748 | 10771 | 10772 | 10773 |
| | 10774 | 10776 | 11055 | | | | | | | | | | | | |
| \$THEN | 6027# | 8906 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 | 9122 | 9133 |
| | 9145 | 9165 | 9192 | 9231 | 9247 | 9249 | 9281 | 9307 | 9325 | 9350 | 9368 | 9395 | 9413 | 9414 | 9449 |

| | | | | | | | | | | | | | | | |
|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| | 9460 | 9474 | 9498 | 9507 | 9568 | 9606 | 9643 | 9689 | 9720 | 9727 | 9728 | 9729 | 9739 | 9758 | 9792 |
| | 9814 | 9815 | 9816 | 9819 | 9822 | 9854 | 9879 | 9886 | 9918 | 9921 | 9940 | 9944 | 9947 | 9949 | 9951 |
| | 9956 | 9968 | 9971 | 9974 | 9979 | 10015 | 10034 | 10035 | 10038 | 10052 | 10103 | 10126 | 10128 | 10181 | 10376 |
| | 10392 | 10394 | 10415 | 10452 | 10467 | 10471 | 10483 | 10485 | 10492 | 10496 | 10522 | 10549 | 10576 | 10603 | 10714 |
| | 10721 | 10738 | | | | | | | | | | | | | |
| \$STILA | 6526# | | | | | | | | | | | | | | |
| \$STILO | 6563# | | | | | | | | | | | | | | |
| \$SUNTL2 | 6621# | 9808 | 10479 | 10490 | | | | | | | | | | | |
| \$SUNTL3 | 6600# | | | | | | | | | | | | | | |
| \$SWHILE | 6460# | | | | | | | | | | | | | | |
| \$SSCMRE | 8857# | | | | | | | | | | | | | | |
| \$SSCMTM | 8857# | | | | | | | | | | | | | | |
| \$SDEFB | 7354# | | | | | | | | | | | | | | |
| \$SENDS | 7305# | | | | | | | | | | | | | | |
| \$SERRO | 6012# | | | | | | | | | | | | | | |
| \$SESCE | 1711# | 8724# | | | | | | | | | | | | | |
| \$SGEN | 5888# | 8908 | 8910 | 8934 | 8944 | 8994 | 9002 | 9012 | 9023 | 9035 | 9058 | 9068 | 9079 | 9091 | 9115 |
| | 9125 | 9136 | 9148 | 9172 | 9194 | 9197 | 9218 | 9237 | 9245 | 9246 | 9254 | 9255 | 9257 | 9259 | 9281 |
| | 9283 | 9299 | 9310 | 9325 | 9327 | 9343 | 9354 | 9368 | 9370 | 9386 | 9399 | 9416 | 9418 | 9419 | 9421 |
| | 9438 | 9455 | 9468 | 9480 | 9496 | 9505 | 9511 | 9562 | 9571 | 9606 | 9608 | 9638 | 9646 | 9670 | 9689 |
| | 9692 | 9695 | 9705 | 9715 | 9723 | 9733 | 9734 | 9738 | 9739 | 9744 | 9745 | 9758 | 9760 | 9794 | 9796 |
| | 9807 | 9808 | 9819 | 9822 | 9825 | 9828 | 9829 | 9832 | 9833 | 9857 | 9879 | 9884 | 9888 | 9891 | 9892 |
| | 9918 | 9920 | 9924 | 9925 | 9938 | 9943 | 9946 | 9954 | 9955 | 9958 | 9959 | 9960 | 9970 | 9973 | 9978 |
| | 9984 | 9996 | 10012 | 10018 | 10037 | 10041 | 10042 | 10050 | 10055 | 10079 | 10089 | 10102 | 10119 | 10120 | 10130 |
| | 10131 | 10188 | 10349 | 10356 | 10375 | 10385 | 10387 | 10396 | 10397 | 10414 | 10418 | 10421 | 10430 | 10435 | 10451 |
| | 10466 | 10469 | 10473 | 10479 | 10480 | 10488 | 10489 | 10490 | 10495 | 10499 | 10527 | 10530 | 10554 | 10557 | 10581 |
| | 10584 | 10608 | 10611 | 10704 | 10712 | 10716 | 10718 | 10724 | 10732 | 10740 | 10748 | 10768 | 10771 | 10772 | 10773 |
| | 10774 | 10776 | 11042 | 11055 | | | | | | | | | | | |
| \$SGETS | 5851# | 8908 | 8910 | 8944 | 8948 | 8994 | 9002 | 9012 | 9023 | 9035 | 9058 | 9068 | 9079 | 9091 | 9115 |
| | 9125 | 9136 | 9148 | 9172 | 9194 | 9218 | 9237 | 9245 | 9246 | 9254 | 9255 | 9257 | 9258 | 9259 | 9283 |
| | 9299 | 9310 | 9327 | 9343 | 9354 | 9370 | 9386 | 9399 | 9416 | 9418 | 9419 | 9421 | 9438 | 9455 | 9468 |
| | 9480 | 9496 | 9505 | 9511 | 9562 | 9571 | 9608 | 9638 | 9646 | 9670 | 9692 | 9695 | 9705 | 9715 | 9723 |
| | 9733 | 9734 | 9738 | 9739 | 9744 | 9745 | 9760 | 9794 | 9796 | 9808 | 9819 | 9822 | 9825 | 9828 | 9829 |
| | 9832 | 9833 | 9857 | 9884 | 9888 | 9891 | 9892 | 9920 | 9924 | 9925 | 9938 | 9943 | 9946 | 9954 | 9955 |
| | 9958 | 9959 | 9960 | 9970 | 9973 | 9978 | 9984 | 9996 | 10012 | 10018 | 10037 | 10041 | 10042 | 10050 | 10055 |
| | 10081 | 10093 | 10119 | 10120 | 10130 | 10131 | 10132 | 10188 | 10351 | 10363 | 10385 | 10387 | 10396 | 10397 | 10398 |
| | 10418 | 10421 | 10423 | 10431 | 10437 | 10469 | 10473 | 10479 | 10480 | 10488 | 10489 | 10490 | 10495 | 10499 | 10527 |
| | 10530 | 10554 | 10557 | 10581 | 10584 | 10608 | 10611 | 10716 | 10718 | 10724 | 10726 | 10732 | 10740 | 10748 | 10771 |
| | 10772 | 10773 | 10774 | 10776 | 11055 | | | | | | | | | | |
| \$SGETT | 5861# | 8908 | 9237 | 9255 | 9258 | 9416 | 9419 | 9734 | 9739 | 9794 | 9819 | 9822 | 9825 | 9829 | 9884 |
| | 9888 | 9920 | 9955 | 10037 | 10385 | 10418 | 10527 | 10554 | 10581 | 10608 | 10716 | 10726 | | | |
| \$SLPCN | 6887# | 9695 | 10771 | 10772 | | | | | | | | | | | |
| \$SNEW | 1662# | 8724# | 8923 | 8986 | 9044 | 9104 | 9157 | 9186 | 9275 | 9319 | 9362 | 9407 | 9528 | 9601 | 9684 |
| | 9753 | 9910 | 10029 | 10070 | 10338 | | | | | | | | | | |
| \$SPOP | 5875# | 8908 | 8910 | 8944 | 8948 | 8994 | 9002 | 9012 | 9023 | 9035 | 9058 | 9068 | 9079 | 9091 | 9115 |
| | 9125 | 9136 | 9148 | 9172 | 9194 | 9218 | 9237 | 9245 | 9246 | 9254 | 9255 | 9257 | 9259 | 9283 | 9299 |
| | 9310 | 9327 | 9343 | 9354 | 9370 | 9386 | 9399 | 9416 | 9418 | 9419 | 9421 | 9438 | 9455 | 9468 | 9480 |
| | 9496 | 9505 | 9511 | 9562 | 9571 | 9608 | 9638 | 9646 | 9670 | 9692 | 9695 | 9705 | 9715 | 9723 | 9733 |
| | 9734 | 9738 | 9739 | 9744 | 9745 | 9760 | 9794 | 9796 | 9808 | 9819 | 9822 | 9825 | 9828 | 9829 | 9832 |
| | 9833 | 9857 | 9884 | 9888 | 9891 | 9892 | 9920 | 9924 | 9925 | 9938 | 9943 | 9946 | 9954 | 9955 | 9958 |
| | 9959 | 9960 | 9970 | 9973 | 9978 | 9984 | 9996 | 10012 | 10018 | 10037 | 10041 | 10042 | 10050 | 10055 | 10081 |
| | 10093 | 10119 | 10120 | 10130 | 10131 | 10132 | 10188 | 10351 | 10363 | 10385 | 10387 | 10396 | 10397 | 10398 | 10418 |
| | 10421 | 10423 | 10431 | 10437 | 10469 | 10473 | 10479 | 10480 | 10488 | 10489 | 10490 | 10495 | 10499 | 10527 | 10530 |
| | 10554 | 10557 | 10581 | 10584 | 10608 | 10611 | 10716 | 10718 | 10724 | 10732 | 10740 | 10748 | 10771 | 10772 | 10773 |
| | 10774 | 10776 | 11055 | | | | | | | | | | | | |
| \$SPUSH | 5839# | 8906 | 8908 | 8934 | 8941 | 8992 | 8999 | 9009 | 9020 | 9032 | 9055 | 9065 | 9076 | 9088 | 9112 |

| | | | |
|---------|-------|-------|-------|
| .\$SCOP | 2454# | 8609# | 11078 |
| .\$SIZE | 4361# | | |
| .\$SUPR | 4913# | | |
| .\$TRAP | 4073# | 8609# | 11087 |
| .\$TYPB | 3287# | | |
| .\$TYPD | 3209# | 8609# | 11081 |
| .\$TYPE | 2985# | 8608# | 11066 |
| .\$TYPO | 3112# | 8610# | 11084 |
| .\$4OCA | 972# | | |

. ABS. 017274 000

ERRORS DETECTED: 0

CNDLAA,CNDLAA/CRF/NL:TOC=CNMAC2.SML,CNMAC.MAC,CNDLAA.P11
RUN-TIME: 80 82 5 SECONDS
RUN-TIME RATIO: 374/168=2.2
CORE USED: 42K (84 PAGES)